ESD-TR-90-103

Annual Report

# Knowledge-Based System Analysis and Control

30 September 1989

**Lincoln Laboratory**

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

*LEXINGTON, MASSACHUSETTS*

**Prepared for the Department of the Air Force under Contract F19628-90-C-0002.**

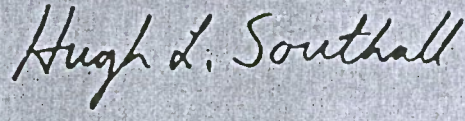
This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. The work was sponsored by the Department of the Air Force under Contract F19628-90-C-0002.



The ESD Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER



Hugh L. Southall, Lt. Col., USAF
Chief, ESD Lincoln Laboratory Project Office

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

# KNOWLEDGE-BASED SYSTEM ANALYSIS AND CONTROL

ANNUAL REPORT SUBMITTED TO
CAPT. ARLAN MORSE
RADC/DCLD
GRIFFISS AFB, NY 13441

*H.M. HEGGESTAD*
*Group 21*

1 OCTOBER 1988 — 30 SEPTEMBER 1989

ISSUED 15 AUGUST 1990



LEXINGTON MASSACHUSETTS

## ABSTRACT

The Machine Intelligent Tech Controller (MITEC) is a software system that operates upon a hardware testbed. The latter is a set of communications, switching and test equipment representing the state of the art as typically seen in modern commercial communications centers. As such these equipment items are typical of new equipment that would be bought and installed to modernize Tech Control Facilities (TCFs) throughout the Air Force if funds were budgeted for this purpose. In particular, they are remotely controllable from a computer or terminal. In the MITEC testbed all the equipment control lines are connected to the MITEC computer, and the MITEC software performs circuit fault isolation and service restoral by directly controlling the equipment, without human intervention.

Two MITEC systems are in place and undergoing further development at Lincoln Laboratory. In operation the two expert systems collaborate with each other much as skilled human Tech Controllers do: the first one to become aware of a circuit fault initiates the diagnosis, while the other provides test signals and measurements upon request. Should the first MITEC determine that the fault is actually on the other MITEC's premises, control will be handed over and the second system will proceed with the diagnosis. The design and operation of the MITEC software and hardware is described in the body of the report.

Plans are developing to deploy two additional MITEC systems in the Washington area for demonstration purposes in the coming year. Another recent extension is commencement of preparations for TRAMCON (Transmission Monitoring and Control) system alarm interpretation and correlation. Work in both of these areas is described.

TABLE OF CONTENTS


| | | |
|---|---|---|
| Abstract | | iii |
| List of Illustrations | | ix |
| List of Tables | | ix |
| 1.0 EXECUTIVE SUMMARY AND INTRODUCTION | | 1 |
| 1.1 | Executive Summary | 1 |
| 1.2 | Background | 3 |
| 1.3 | The MITEC Design Concept | 5 |
| 1.4 | Project Chronology | 5 |
| 1.4.1 | The Expert Tech Controller (ETC) | 5 |
| 1.4.2 | The Machine Intelligent Tech Controller (MITEC) | 7 |
| 2.0 MITEC SYSTEM CONCEPT | | 11 |
| 2.1 | System Overview | 11 |
| 2.2 | Testbed Architecture and Hardware Selection | 13 |
| 2.3 | The Ordered List of Capabilities | 14 |
| 2.4 | Fault Diagnosis/Restoration Strategy Issues | 16 |
| 3.0 HARDWARE TESTBED | | 19 |
| 3.1 | MITEC Testbed | 19 |
| 3.2 | Testbed Components | 22 |
| 3.2.1 | Basic Communications | 22 |
| 3.2.2 | Patching and Circuit Accessing | 23 |
| 3.2.3 | Alarm Sensing | 28 |
| 3.2.4 | Signal Measurement | 28 |
| 3.2.5 | Fault Insertion, Line Degradation | 31 |
| 3.3 | Testbed Diagram | 33 |
| 4.0 MITEC SYSTEM ORGANIZATION | | 37 |
| 4.1 | System Architecture | 37 |
| 4.1.1 | Symbolics Computer | 37 |
| 4.1.2 | Sun Computer | 37 |
| 4.1.3 | Communication with Devices | 39 |
| 4.1.3.1 | Philips Digital Storage Oscilloscope PM3352 | 40 |
| 4.1.3.2 | Datalok 10A with Relay Scanner | 40 |
| 4.1.3.3 | TELENEX Mini-Matrix Switch | 40 |
| 4.1.3.4 | AN/FCC-100 Multiplexer | 41 |
| 4.1.3.5 | Hekimian 3701, 3703, 3705, 3200 | 42 |
| 4.2 | SCHEDULING SYSTEM | 42 |
| 4.2.1 | Scheduling of Resources | 42 |
| 4.2.1.1 | Process Scheduling | 42 |
| 4.2.1.2 | Device Scheduling | 43 |
| 4.2.1.3 | Scheduler Window | 44 |
| 4.2.2 | Communication | 45 |

TABLE OF CONTENTS (CONTINUED)


4.2.2.1 Process to Process 45
4.2.2.2 Process to Device 47
4.2.2.3 MITEC to MITEC 47
4.2.3 Operator Interface 48
4.2.3.1 Terminal Interactions 48
4.2.3.2 Panes 49
4.2.3.3 Graphics 49
4.2.3.4 Modes and Options 50
4.2.4 Operator Interface Applications 50
4.2.4.1 Browsing 50
4.2.4.2 Diagnosis 51
4.2.5 Database 54
4.2.5.1 Purpose 54
4.2.5.2 Organization 54
4.2.5.3 Implementation 56
4.2.5.4 Issues 63
4.3 MITEC Fault Isolation and Restoral 64
4.3.1 Fault Isolation Phases 65
4.3.1.1 Trouble Detection 65
4.3.1.2 Troubleshooting and Restoral Strategy 66
4.3.2 Troubleshooting Circuits Between MITECs 67
4.3.3 Troubleshooting Tail-Circuits 68
4.3.4 Signal Quality Assessment: Oscilloscope for Digital Signals 70
4.3.4.1 Introduction 70
4.3.4.2 Waveform Parameters 70
4.3.4.3 Waveform Setup 73
4.3.4.4 Waveform Analysis 73
4.3.4.5 Waveform Display 74
4.3.4.6 Waveforms for Later Use 75
4.3.5 Signal Quality Assessment of VF Signals 75
4.3.6 Observations 75

5.0 TRAMCON/DPAS ALARM INTEGRATION 79

5.1 Introduction 79
5.2 Specification for TRAMCON and DPAS Alarm Generators 79
5.2.1 Background 79
5.2.2 Microwave Radio System Description 81
5.2.3 TRAMCON Operation 81
5.2.4 DPAS Description 81
5.2.5 Potential MITEC Interactions with TRAMCON and DPAS 82
5.2.6 TRAMCON and DPAS ALarm Generator Requirements 83
5.2.7 Development Sequence 83

TABLE OF CONTENTS (CONTINUED)


| | | |
|---|---|---|
| APPENDIX A | MITEC Equipment List | 85 |
| APPENDIX B | Device Interface Requirements | 89 |
| APPENDIX C | Device Communication Experiences | 101 |
| APPENDIX D | TRAMCON Alarm Generator Design Requirements | 105 |
| APPENDIX E | Pentagon - Ft. Detrick Demonstration Plans | 151 |
| REFERENCES | | 153 |

## LIST OF ILLUSTRATIONS


| FIGURE NO. | | PAGE |
|---|---|---|
| Figure 2.1 | MITEC Testbed Block Diagram | 12 |
| Figure 3.1 | MITEC in a Tech Control Environment | 20 |
| Figure 3.2 | Testbed Communications Equipment and Circuits | 24 |
| Figure 3.3 | DPAS/DACS II in the Testbed | 29 |
| Figure 3.4 | West TC Diagram | 34 |
| Figure 4.1 | Scheduler Monitor Window for TCF: West | 46 |
| Figure 4.2 | CDMO (Circuit) | 53 |
| Figure 4.3 | MITEC Display During a Diagnosis | 55 |
| Figure 4.4 | Device Feature Hierarchy | 57 |
| Figure 4.5 | Circuit Feature Hierarchy | 58 |
| Figure 4.6 | Circuit CDMO Database | 59 |
| Figure 4.7 | Trunk BBTDMO Database | 60 |
| Figure 4.8 | FCC-100-TDMO Database | 61 |
| Figure 4.9 | CODEX-2400-4W-CDM0-1 Database | 62 |
| Figure 4.10 | CDMO Circuit Layout Display | 69 |
| Figure 4.11 | Tail Circuit Display | 71 |
| Figure 4.12 | Waveform Display | 76 |
| Figure 5.1 | Alarm Monitoring Systems | 80 |


## LIST OF TABLES


| TABLE NO. | | PAGE |
|---|---|---|
| Table 1 | Language Processors Suitable for TEG | 112 |

# 1.0 EXECUTIVE SUMMARY AND INTRODUCTION

This chapter is provided in order to give a brief overview of the progress and technical content reported in the main body of the document. It also includes a brief review of the background and a more detailed description of the chronology of the project. Together these sections provide an introduction for the report.

## 1.1 Executive Summary

The Machine Intelligent Tech Controller (MITEC) is a software system that operates upon a hardware testbed. The software is an expert system which performs the communications circuit fault isolation and service restoral functions currently done by staffs of Tech Controllers at military Tech Control Facilities (TCFs) worldwide. Specifically, the functions of MITEC are to:

1. Become aware of the existence of a circuit degradation or failure, either by observing an alarm or by receiving notification from a human operator;

2. Follow a logical procedure to isolate and identify the faulty segment of the circuit;

3. Search the data base for spare or preemptable assets to substitute for the faulty segment;

4. Electronically switch the replacement assets into the circuit, thus restoring service; and

5. Report the results of the action to the human operator for relay to higher authority, and for notification of repair personnel to localize and fix the specific fault that caused the outage.

Chapter 2.0 of this report provides a detailed description of the MITEC system concept, including explanations of the above functions.

The operation of the MITEC software is inseparable from the operation of the associated communications, switching and test equipment. In fact, the essence of the MITEC design philosophy is autonomous execution of the entire fault isolation/service restoral process, without need for human intervention. MITEC is aimed at communication equipment one or more generations more modern than that in today's TCFs, which is typically decades-old technology that is strictly manually-operated. Thus it was necessary to design and build a scaled-down testbed version of a modern TCF, with a few representative circuits and one or more examples of each of an interesting variety of equipment types.

In developing the MITEC testbed, information was obtained from a variety of sources including commercial communication centers, manufacturers' catalogs and representatives, and contacts in the Tech Control community. The body of this report describes the semiannual MITEC Steering Group process in which senior Tech Controllers were invited to review and approve, or change, equipment selections and functionality of the testbed.

Since a circuit outage typically involves two or more TCFs, the minimum configuration for the MITEC project was two stations. Accordingly two have been built and installed at Lincoln Laboratory, and they have trunk circuits joining them as well as communities of end users at each testbed. The communications, switching and test equipment in each testbed is connected to a Sun workstation which serves as a multiplexer to share the two ports of the MITEC computer (a Symbolics 3650) among all the equipment. Chapter 3.0 of this report gives a complete description of the design and operation of the MITEC testbeds.

The MITEC software is implemented in ZetaLISP, the native LISP version of the Symbolics computer. The main components of the software architecture include extensive facilities for communicating with the devices in the testbed; a scheduling system for managing the allocation of software and hardware resources and the operator interface facilities; and the algorithmic software structures that carry out the various operations in troubleshooting and restoral. The design and implementation of the software is described in detail in Chapter 4.0 of this report.

The transmission facilities in the Defense Communications System (DCS) are moving more and more toward exclusive reliance upon multi-channel digital T1 carrier systems, typically carried by microwave radio links and interconnected by means of Digital Patch and Access System (DPAS) electronic switches. The Transmission Monitoring and Control (TRAMCON) alarm polling system is becoming widely deployed among microwave radio sites overseas, and the information gathered by TRAMCON can be of great help in troubleshooting circuits. At the same time, the DPAS switches generate alarm signals of various kinds when the T1 carriers feeding them develop problems. It has been recognized that a valuable adjunct to MITEC operation would be a capability to evaluate, correlate and understand patterns of TRAMCON and DPAS alarms, so that the resulting information can be made available to strengthen the fault isolation algorithms. Chapter 5.0 of this report describes progress during the current year in specifying and designing a system that simulates the alarm generation behavior of a communications system, and describes how this simulator will be used as a testbed upon which to develop expert knowledge on interpretation of complex alarm patterns.

The Appendices to this report include a list of the details of the MITEC testbed equipment; a set of specifications we have evolved for equipment remote control ports, such that an external computer (such as MITEC) could successfully interface with them; a summary of our experiences in connecting to equipment; the details of the TRAMCON Event Generator specification; and plans for the Washington-area demonstrations of two MITECs in FY90.

## 1.2 Background

MITEC has its evolutionary origins in the Expert Tech Controller (ETC) project [1,2,3,4]. The ETC project was based upon an earlier study of Tech Control automation techniques by Rome Air Development Center (RADC/DCLD), and upon a study by Lincoln Laboratory on the potential applications of artificial intelligence technology in military communications. RADC/DCLD funded the ETC project, continuing as it evolved into MITEC, and is currently sponsoring field demonstrations of MITEC which are expected to lead to technology transfer of MITEC into the operational Tech Control inventory. Another aspect of the expected future evolution of MITEC will be interfacing to a distributed expert system structure for communications control, a new RADC-sponsored development effort at Lincoln.

The ETC project was conducted by Lincoln Laboratory in FY86-FY87 as a proof-of-concept demonstration that Tech Controller skills and knowledge could be captured in an expert system and made continually available to junior personnel so that they can effectively deliver performance well above their skill level. The ETC architecture features an "air gap" between its reasoning processes and the Tech Control environment: ETC is completely contained within a Symbolics computer, and depends upon its human operator to make measurements, provide inputs and carry out instructions. In contrast, MITEC interacts directly with its outside world, so that the human can be out of the loop unless he chooses to be involved.

This direct-interaction mode of operation is of course not yet possible for MITEC in a typical present-day Tech Control Facility (TCF) with its older hand-operated patch panels and equipment. MITEC anticipates the evolutionary upgrading of TCFs to an environment of modern remotely-operable communications, test and electronic switching equipment which can be queried and controlled via electrical links from a terminal or computer. Thus MITEC can provide fault isolation and service restoral capabilities that are truly independent of human participation for a wide range of trouble types. Alternatively, options are available for the human operator to check/approve MITEC actions in a lockstep mode, or to take over a diagnosis altogether.

ETC realized its primary system design goal, which was to be able to accomplish fault isolation on a number of circuit types

commonly found in TCFs, for a variety of typical faults and symptoms, using logic and procedures recommended by practicing Tech Controllers. To use ETC, an operator enters the CCSD identifier and the observed symptoms of a failed circuit via keyboard and mouse, and ETC conducts a dialog with the operator as it requests and receives the successive pieces of information it needs in the diagnosis. Typically the questions involve signal measurements or verification of signal presence/absence at a sequence of locations, and ETC instructs the operator as to where to go and what to do to obtain the answers. Upon reaching a conclusion as to the nature of the fault, ETC instructs the operator on how to access spares or alternate paths to restore service if possible, and how to initiate repair of the failed components.

ETC provides an interesting demonstration and is very effective in showing people the nature and value of an expert system in the Tech Control context. It also has great potential for use as a training tool (given considerable enlargement of its present circuit data base and problem repertoire), because it can provide step-by-step demonstration of fault isolation techniques for all varieties of typical problems on the circuits in its data base. ETC also provides a flexible "what-if" capability: besides supporting instruction in the basics for junior personnel, ETC can provide experimentation and practice opportunities for personnel at all skill levels for complex and important problem scenarios that may occur so seldom that real-world experience is rare.

As a tool for day-to-day assistance in Tech Control operations, however, ETC as it presently exists has major drawbacks. An obvious issue is the need for the human operator to fetch and carry as directed by the system; as soon as the operator has skills beyond a rather basic level he will be capable of solving many problems quickly and independently, and will become impatient with ETC's lockstep mode of operation. (And note that ETC requires this lockstep operation, without which its reasoning processes will be in error because of assuming conditions or completed steps that have not in fact been done.)

A more subtle but very important issue is that, in order for ETC to perform correctly in live operation, it must be kept informed of all changes and Tech Controller actions (such as circuit patches or equipment availability changes) that affect the correctness of ETC's station data base. This would quickly become a burden for operations personnel.

Thus it was clearly evident that the "air gap" between ETC and the TCF equipment would have to be closed if expert system technology is to be practically applied in a TCF environment. Largely as a result of the successful implementation and

demonstration of ETC, support was obtained for a new phase of the work aimed at closing the air gap.

## 1.3 The MITEC Design Concept

As explained more fully in Chapter 2.0, a major aspect of the development of MITEC is implementation of a full set of capabilities to control modern communications and test equipment representative of that which may be acquired and installed at TCFs during modernization programs. Another major requirement is to evaluate and understand all the functional capabilities of such equipment with respect to fault isolation and service restoral, some of which may be expected to substantially transcend the capabilities of the older, conventional manual equipment contemplated by ETC. Clearly it was necessary to launch a program of study and investigation of this equipment if the MITEC goals were to be realized.

This led to four interlocking problems:

1. How to identify the equipment types and items with which MITEC should operate;

2. How to obtain complete information on the interface/control characteristics of this equipment so as to define the MITEC side of the control interfaces;

3. How to create and capture fault isolation/service restoral knowledge exploiting the equipment capabilities; and

4. How to test and validate the resulting MITEC code.

It quickly became clear that the most expeditious means to resolve this difficult set of problems was to procure and assemble actual communications and test equipment, physically interface it to MITEC and by careful study and experimentation, to work out techniques for using the equipment to accomplish the TCF missions. A plan was developed and executed to achieve these objectives, as described in more detail in the following section.

## 1.4 Project Chronology

### 1.4.1 The Expert Tech Controller (ETC)

During FY85 a study was carried out by Lincoln Laboratory to identify promising applications of Artificial Intelligence technology in military communications. Attention shortly began to focus upon Expert Systems, as an emerging area of AI technology which appeared to offer a practical remedy for a serious problem common throughout the military establishment, namely preserving longer-term benefits from the slow and expensive process of training personnel in complex professional

specialties. Tech Control began to emerge as a skill area ripe for automation, as strongly recommended by RADC/DCLD. Lincoln staff visited a number of Army, Navy and Air Force TCFs to learn about their particular problems and to evaluate them in terms of the knowledge engineering and rule development paradigms of expert systems technology. The culmination of the study was a presentation in the summer of 1985 to RADC/DCLD, which subsequently funded an FY86 project at Lincoln Laboratory to begin development of what came to be called ETC. The FY86 Annual Report [2] described the initial build of ETC, which was based upon the ART (tm) expert system shell developed by Inference Corporation, Inc. of Los Angeles. The implementation involved numerous knowledge engineering interactions with the 2045th Communications Group, a major Air Force TCF located at Andrews Air Force Base, Maryland. Early in the project it was found expedient to install a Symbolics 3650 computer at Andrews to host successive versions of the ETC software, so that the correctness of each new increment of fault isolation knowledge could be verified by exposing it directly to the knowledge sources for experimentation and evaluation.

RADC sponsorship of the ETC project was renewed in FY87, with the primary goal of developing the system to the point of enabling transfer of the technology to a fielded system. It was recognized that three key ingredients were needed: (1) demonstration of a working ETC prototype having a substantial repertoire of Tech Control knowledge and capability, (2) development of substantial support from the user community, in terms of both encouragement and funding, and (3) production of system documentation with sufficient detail and completeness to serve as the technical core of a specification for follow-on procurement of the field-deployable re-implementation of ETC. Demonstrations of the current version of ETC were given quite often, both at Lincoln Laboratory and at Andrews AFB. It was recognized that a live demonstration of ETC was far more effective than a briefing for imparting an understanding of what the system can do, but unfortunately a trip to one of the demo sites was impractical for many people who should be exposed to the technology (such as staff and management at the Air Force Communications Command). Accordingly a video tape was produced, giving an introduction to Tech Control functions and sequences of narrated ETC screen display sequences that had nearly the impact of a live demonstration. This tape was copied and distributed to a number of interested parties, and contributed to the growth of support for the MITEC project as discussed below.

Early in FY87 it was recognized that the operation of ETC was becoming more and more sluggish as its knowledge base was increased. For example, it was taking as long as 20 minutes to do the ART reset that was necessary after each diagnosis operation. The system was carefully analyzed, and it was concluded that the problems were largely with ART itself, and

were related to the overhead of making ART a highly general tool for many kinds of problems. Accordingly the development of ETC was temporarily suspended for about a month, while the entire system was rewritten in Symbolics LISP. The effort was successful: the reset problem disappeared, system operation speeded up satisfactorily, and system knowledge base growth since that time has caused no perceptible slowdown. The experience paralleled those reported by many developers of expert systems in recent years: an initial system is rapidly prototyped and exploited as a learning tool, then it is essentially thrown out and replaced by a second build which is completed much faster than the first.

### 1.4.2 The Machine Intelligent Tech Controller (MITEC)

Halfway through FY87 it began to appear that the goals of demonstrating an interesting level of ETC capability and writing up its specification could be completed sometime during FY88, and accordingly it was decided by the sponsor to forego an FY87 Annual Report on the grounds that the effort could be better spent on writing the specification document as soon as it would be feasible to do so. Concurrently, however, the reactions and advice of numerous Tech Control professionals began to make it increasingly clear that the "air gap" inherent in ETC was going to effectively prevent the system from ever achieving significant usefulness as an operational tool in Tech Control. The problems noted in Section 1.0 of this report -- primarily the need to keep ETC up to date on every patching and diagnosis action in the facility, including those which are done on human initiative without need for guidance from ETC -- would make it unattractive for the users to apply ETC for any purpose but offline training.

In late FY87 and early FY88, planning was therefore begun for a new expert system which would have no air gap, but would directly control equipment as outlined in Sections 1.0 and 2.0 of this report. Procurement of equipment to outfit the West and East TCFs was initiated in FY88, and by June 1988 the design was well enough in hand to conduct the first semiannual MITEC Steering Group meeting. The invitees included Tech Control professionals from AFCC, SAC, and the Air Force Logistics Command, at both senior NCO and management levels. The meeting was held at Lincoln Laboratory on 21-23 June, and it resulted in crystallizing the design and the objectives of the MITEC testbed and the goals of the system software.

At the end of FY88, it was agreed by the sponsor that it made better sense to postpone the writing effort that would have gone into an FY88 Annual Report, in favor of pushing toward completion of the MITEC software and testbed and documenting them fully when ready.

The second semiannual MITEC Steering Group meeting was held at Lincoln Laboratory in the period 6-8 December 1988. Besides the organizations represented at the June Meeting, the Air Force 7th Comm Group (at the Pentagon) was invited to attend. It was agreed that Lincoln Laboratory would wrap up the ETC project by writing an ETC User's Manual, to be ready by about March 1989. The ordered list of MITEC capabilities was reviewed and discussed, and Lincoln progress toward achieving them was reviewed. The support of AFCC was requested as a key to transfer of the MITEC technology into the Air Force inventory, and it was agreed that AFCC would contribute substantial funding in FY90 and beyond. An important discussion topic was the upcoming DCA-sponsored Tech Control Automation Proof of Concept System (TCAPS) demonstation, which was to include replicas of MITEC doing automated fault isolation; it was agreed that besides the primary TCAPS site (Ft. Detrick, MD) the 7th Comm Group would host a MITEC for this demo.

The third MITEC Steering Group meeting was held at Lincoln Laboratory in the period 18-20 July 1989. Considerable effort went into preparing a preliminary demonstration of fully-automated fault isolation by separated independent MITECs on the two-TCF testbed; this was to be a precursor of formal MITEC demos originally scheduled for the end of FY89. The preliminary demos were in fact very successful, and indeed it was decided that the purpose planned for the fall demos had already been served. For example, in troubleshooting a fault and restoring a circuit on an LSTDM port pair between the two testbed sites, the MITECs accomplished the whole task in 27 seconds. Senior Tech Controllers present estimated that manual resolution of a similar problem would typically take TCF staff about 30 minutes. There was much additional discussion of the TCAPS demos, and the DCA gave a report on their procurement of copies of MITEC testbed hardware for installation at the two TCAPS sites.

At the 18-20 July meeting Lincoln delivered a draft of an import part of the MITEC documentation under preparation, namely a System/Segment Specification. A specification was described for a TRAMCON/DPAS Event Generator which would emulate the generation of transmission system alarms as inputs to the expert system processes of MITEC, during design and test of MITEC software.

In the intervening period up to the present, a steadily increasing emphasis has been placed upon the MITEC demonstrations as part of TCAPS. It is anticipated that these demos can lead to a go/no-go decision on whether to proceed with transfer of the MITEC technology into the field. Alternatives are being considered for undertaking such transfer; the most reasonable approach appears to be to use contract software engineering staff to re-implement the MITEC software in Ada, following military software design and development standards throughout, and aiming

at a Unix environment which can potentially be deployed in the field at modest cost.

## 2.0 MITEC SYSTEM CONCEPT

### 2.1 System Overview

Figure 2-1 is a top-level block diagram of the MITEC Testbed system at Lincoln Laboratory (details are described in the Section 3.0 of this report). The West and East TCFs each represent a modern Tech Control Facility outfitted with communications, test and electronic switching equipment meeting the remote control requirements for MITEC. Instead of dozens or hundreds of copies of the various equipment types as in a real TCF, the MITEC testbed systems have one or two each of an interesting array of representative items. As shown in Fig. 2-1, each TCF has a community of users associated with it; these are one each of a set of representative voice and data users such as one might find in the real-world environment, with connectivity through the TCFs to their counterparts at the far end. The East and West TCFs are linked by transmission media including T1 and VF circuits. The East and West MITECs communicate with each other by means of the orderwire illustrated in the figure.

Each TCF has a MITEC provided with control and alarm interfaces to its associated testbed equipment. In operation, it is expected that the MITEC demonstration director will create a fault somewhere, either by disabling a selected port or card within a TCF or by degrading a transmission link between East and West. The nearest MITEC will be notified of the outage, either by noting an alarm signal that it monitors, or by receiving an (emulated) user complaint. The alerted MITEC will then proceed to follow a logical sequence of steps to isolate and restore the failed segment of the circuit. These steps may involve requesting test signals and other services from the other MITEC, and they may also involve concluding that the fault location is actually at the other TCF, whereupon the fault isolation responsibility is automatically transferred to the other MITEC.

Ultimately the fault location is identified, and the MITEC in charge proceeds to check its database for spare paths or equipment to exploit in restoring service. If spares are available, the MITEC in charge requests the collaboration of its counterpart at the other TCF (if necessary) to implement the necessary patches to place the spares in service. For example, if the problem was found to be a bad channel somewhere on a pair of multiplexers, the fix may be to switch the affected circuit to a spare channel on the same muxes; this can be accomplished only if the patch to the spare channel is done simultaneously at both TCFs.

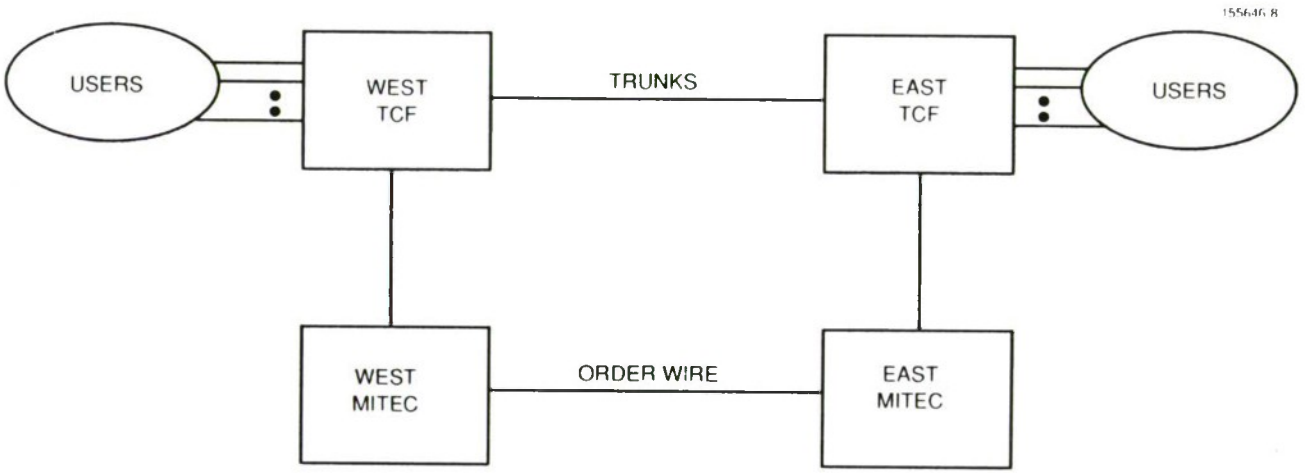



Figure 2.1
MITEC Testbed Block Diagram

## 2.2 Testbed Architecture and Hardware Selection

The commercial telecommunications industry has been developing a wide assortment of modern equipment in pursuit of automation and improved efficiency, and there are various competing products available for each generic application in a TCF. The selection of specific equipment and connection schemes for the MITEC testbed is complicated by the facts that some commercial requirements differ substantially from military needs, and that little of the interesting modern equipment has been selected and installed in real military TCFs as yet. Two approaches have been pursued iteratively in overcoming these complications and making specific selections for the testbed, namely consulting with senior Tech Control specialists and studying the characteristics of available equipment. The latter activity has included numerous discussions with equipment vendors as well as visits and consultations at large-scale commercial telecommunications operations.

The actual selection of equipment items was a compromise in each case among a number of factors:

1. Technical requirement for a particular function -- for example, the recognition that the use of existing VF tail circuits to carry digital data will undoubtedly continue for many years into the future, thus implying that modems should be included.

2. Availability of a suitable modern military item that is widely deployed and in current use -- for example, the AN/FCC-100 low-speed time division multiplexer (LSTDM), whose control and configuration functions can be performed remotely via an RS232 port.

3. Testbed (and future TCF modernization) cost containment -- for example, choosing the lowest-priced candidate item that has the necessary characteristics and control features for a particular application.

4. Conformance with popular standards that will evidently continue in force for the foreseeable future -- for example, inclusion of T1 transmission systems compatible with both commercial and military practices.

5. Provision of capabilities that are not currently in the military TCF catalog, but become highly desirable in the automated environment of MITEC -- for example, an oscilloscope that can transmit digitized waveforms to the MITEC computer for automated analysis of signal quality, and an access switch permitting remotely-controlled connection of test equipment to any of a number of test points pre-wired throughout the TCF.
6. Finally, review by knowledgeable senior Tech Controllers in

positions which involve planning the future of their profession -- for example, staff members at the Air Force Communications Command (AFCC).

This work having been in progress for about a year, the two MITEC testbed systems are now in place and operating at Lincoln Laboratory. An additional system is in the process of procurement and installation at RADC to form a third MITEC testbed node, and two more systems have been procured by the DCA for demonstration at the Pentagon and Ft. Detrick, MD (see Appendix E).

## 2.3 The Ordered List of Capabilities

One of the outcomes of the first semiannual MITEC Steering Group meeting (see Section 6.0) was the development of a priority-ordered list of capabilities that the Tech Control community would want in a fully-developed MITEC deployed in the field. This list was subsequently reviewed at the second and third semiannual meetings, and the current version is reproduced and discussed here.

Four general classes of MITEC functions were specified, with the notion that development efforts would work downward from the top of the list, as far as available resources could carry them. These classes are:

1. VF and data circuit test, diagnosis and restoral via spare or preempted assets;

2. Trunk test, diagnosis and restoral via spare or preempted assets;

3. Equipment testing; and

4. Administration, including reports and trend analysis.

The first two classes embrace the primary business of Tech Control, and begin with extension of ETC capabilities to the MITEC environment. The basic requirements of these classes are already implemented in MITEC, as demonstrated at Lincoln Laboratory in July 1989 (see Section 6.0 below) and planned for the Pentagon/Ft. Detrick demos in early 1990 (see Appendix E). It is expected that on a 2-year time scale (i.e., by the end of FY90) substantial capabilities in these areas will be implemented. The required functions are electronically-selected access to test points, including coordination of far-end access by another MITEC; remote connection and operation of test equipment, including participation as necessary by a remote MITEC; collection and analysis of test results; maintenance and search of a data base of in-use and spare communications assets; and electronically-controlled configuration and patching of devices and circuits. These functions are to be performed by MITEC software in a logical sequence leading to

isolation of circuit and trunk faults followed by service restoral via spares (if available), otherwise by an appropriate message to the operator recommending suitable actions. These functions constitute the real challenge in MITEC expert system software development, and are the first priority for Lincoln work on MITEC.

The first two classes also include the practice of in-service testing, which is monitoring of circuit performance without disrupting normal traffic. In this context MITEC would be able to detect incipient failures and act upon them before actual user complaints are received. No work has been done to date on implementation of such MITEC capabilities. While these functions are of considerable potential value in TCF operation, they can be implemented in straightforward procedural code which is much less of a challenge than the MITEC fault isolation software. Should available Lincoln resources not support the implementation of such code in the FY90 time frame, it would be quite realistic to delegate it to a follow-on contractor who might be tasked by the Government (given adequate interest and funding) to produce a complete field-deployable and maintainable version of MITEC software implemented in accordance with DoD-Std 2167.

Equipment testing, the third class of MITEC functions listed above, means automated verification of correct operation of spare communications assets. The concept also includes substituting spares for on-line equipment items, and then testing the latter. These functions are periodically performed by Tech Controllers in current practice, and automating them would be a straightforward application of some of the MITEC capabilities used in diagnosis and restoral. No work has been done to date on actual implementation of such applications. This is another candidate for delegation to a follow-on contractor.

In administrative functions, the fourth class listed above, significant capabilities have already been demonstrated in ETC. Circuit data base entries were presented to the user for browsing or editing via an on-screen replica of a DD Form 1441 similar to those maintained in a physical card file in a TCF. Diagnosis records and trouble tickets were presented as replicas of DD Forms 1443 and 1445, respectively, and hard copy of any of these forms could be obtained by means of the laser printer attached to ETC. Since the information for these forms was either already stored in ETC, or generated automatically in ETC during each fault isolation/service restoral exercise, the implementation of the forms was a straightforward exercise. Similarly, although such forms are not yet activated in MITEC, no difficulties are anticipated in implementing them for internal storage, hard copy generation (if desired), and electronic transmission to other offices or agencies as appropriate. Likewise, given chronological records of the results of periodic measurements of system parameters, it will be straightforward to implement trend analysis software. All of this work could reasonably be delegated to a

competent follow-on contractor.

## 2.4 Fault Diagnosis/Restoration Strategy Issues

A new student Tech Controller is taught basic fault isolation and circuit restoral procedures in service school, and he goes to his first assignment with a beginning tool kit of knowledge and techniques. As his skill level progresses, he accumulates knowledge about numerous shortcuts and special cases that depart from the methodical procedures taught in school. After several years of experience, he is likely to have a fault isolation philosophy which differs substantially from the training manuals, and indeed senior Tech Controllers can get into long discussions over disagreements among themselves as to how their work should be conducted and taught.

One result of this effect has been that knowledge obtained from one set of senior Tech Controllers, encoded in ETC and MITEC, has occasionally drawn fire when the systems are demonstrated to another set of them. This has ramifications in two areas: training and fully-automatic operation. In either case, clearly one would not want the displayed logic and procedures to be wrong in the sense of being vulnerable to failure to get the right answer in some circumstances. Wherever senior Tech Controllers can identify cases where the expert system logic will fail, the logic must be corrected.

When these personnel disagree on the basis of efficiency or philosophy, on the other hand, the implications for the two areas are quite different. If one is training people to perform manual Tech Controller functions (e.g., in the near-term world in which TCFs do not have automated equipment that could be directly controlled by a MITEC), then it is clearly important to teach the most efficient techniques. To the extent that the advising Tech Controllers argue for substantial improvement in manual operation efficiency by doing so, corrections should be made in MITEC code that impacts training.

In fully-automatic operation, however, the logic processes in MITEC generally operate so fast compared to manual procedures that efficiency improvements tend to be less critical. It is more important that the logic always lead to a correct answer (even if that answer is "This problem has been found to fall outside of the MITEC knowledge base, and should be referred to a senior person.") It is also important that the logic be as transparent as possible, so that it will be readily comprehensible to the future maintainers of the MITEC software.

No matter how long efforts continue to refine the MITEC fault isolation knowledge base, it is probable that there will always be some percentage of complex and unusual problems that MITEC does not know how to handle. The goal for FY90 implementation of MITEC at

Lincoln Laboratory is to be able to deal with a substantial fraction (i.e., >50%) of the Tech Controller workload at a normal TCF, assuming it were fully outfitted with remotely-controllable equipment. It is reasonable to assume that continuing improvements can be made in the course of re-implementing MITEC for field deployment, as well as during the subsequent period of regular MITEC software maintenance and upgrade by a military agency such as AFCC/CCSC.

# 3.0 HARDWARE TESTBED

## 3.1 MITEC Testbed

The purpose of the MITEC Testbed is to provide an environment in which the MITEC system concept can be evaluated and demonstrated. MITEC itself is a combination of computer hardware and software designed to be embedded in future tech controls (TCFs) to support automation of certain tech control functions and to aid tech control personnel in the performance of other functions. Figure 3.1 shows a MITEC in such an environment. Four communication paths to and from MITEC are shown. An additional path must exist between MITEC and its human operators, but that path is not shown in the figure. The order wires go to other MITECs and/or to other non-automated facilities to support cooperative fault isolation, circuit testing, and reporting procedures. The paths to local facilities allow MITEC to access test equipment for measurement purposes, to sense alarms, and to directly access communication equipment for the purpose of querying status and changing configurations in the equipment. While these paths are essential for MITEC to perform TCF functions automatically, they are not usually shown in other diagrams of the Testbed that focus on the communication equipment, transmission lines, and Testbed circuits.

Ideally MITEC would be tested and demonstrated in real TCF environments with a wide variety of communication equipment and enough circuits so that real problems would occur with sufficient frequency to allow the effectiveness of MITEC to be evaluated. The small size of the MITEC development effort does not allow such environments to be used. Instead, we have constructed a testbed at Lincoln Laboratory that has equipment for two very small TCFs each with its independent MITEC hardware and software. The TCFs are connected by trunk lines that support multiplexed user circuits, and each has responsibility for the communication quality and reliability of a set of simulated users who may be either local to the TCF or remotely connected through "tail" circuits. The TCFs are jointly responsible for service on the interconnecting trunks but individually responsible for their own tail circuits. The MITECs in the TCFs are connected by an order wire that allows them to cooperate in diagnosing problems and restoring service.

In choosing equipment for the testbed we have made a number of compromises due to budget and time constraints. Ideally, we would have selected equipment already in use in military TCFs or in the procurement process after careful selection to meet military requirements. In some cases we have been able to use such equipment in the testbed. In other cases, we found that such equipment was either too costly for our budget, or that appropriate equipment with the desired capabilities had not been selected for military use. In those cases we chose equipment having the propertie having the properties needed for MITEC and available at a price within our budget. We make no claim that the particular

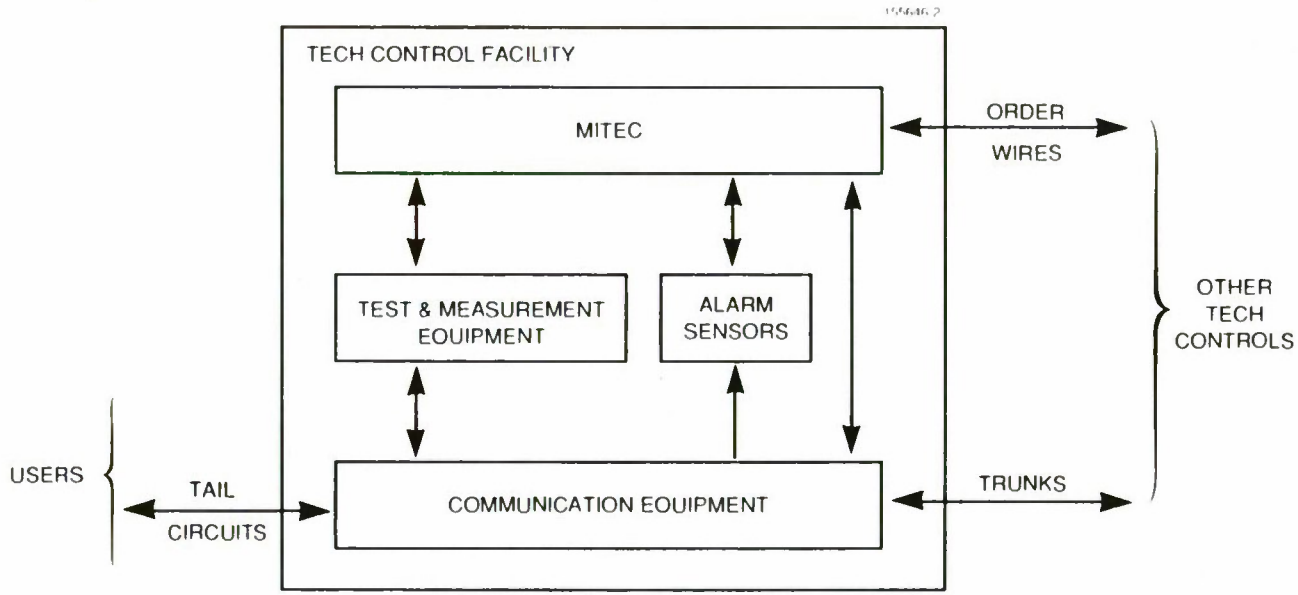



Figure 3.1
MITEC in a Tech Control Environment

equipment we have chosen under these ground rules is appropriate for military use or even for use in a full scale non-military facility.

Some of the testbed communication equipment allows direct control by MITEC. Having such equipment in the testbed is essential for demonstrating fully automated TCF functions. Other equipment in the testbed requires human activity to sense alarm lights or change configurations. By having both kinds we are able to demonstrate that MITEC can deal with an environment in which some "air gap" remains, a situation that would be likely to be found in any real-world application of MITEC.

In examining equipment for possible use in the testbed, we found that manufacturers used different approaches for dealing with the communication paths between their equipment and control and monitoring systems such as MITEC. Some devices were designed to interface to an IEEE-488 bus, others to RS-232 communication lines. A few could cope with both kinds of interface. As to the protocol used, some manufacturers treated it as public information, others as proprietary. We chose to limit MITEC-controllable equipment to devices that use RS-232 communications and that have public protocols.

In addition to the two TCFs and associated MITECs, the Testbed includes simulated transmission lines and users. The transmission lines include both T1 and 4-wire phone line trunks between the TCFs and phone line tail circuits to users. The only user circuits defined are data circuits, and the users of these are simulated by Bit Error Rate Testers (BERTs). The BERTs can generate both synchronous and asynchronous data streams at a wide variety of rates and can readily detect and display errors caused by circuit problems introduced to test MITECs diagnostic and service restoration capabilities.

Voice Frequency (VF) user circuits could easily be added to the testbed, but we decided not to include them because of the difficulty of simulating users in a realistic fashion. There is no problem in providing a telephone through which people could converse, but the human ear is rather tolerant with respect to VF circuit quality. VF circuits are often used as trunk circuits between telephone switches, and when in-band signalling is employed on such trunks, circuit quality issues can become critical. However, we have no convenient means for simulating such use, and we believe that MITEC's VF circuit testing capabilities can be adequately demonstrated on the data circuits and trunks where modems on the phone lines play the roles of VF users, and the BERTs can show the effects of line quality degradations.

## 3.2 Testbed Components

This section enumerates the components of the testbed and discusses each one briefly. They are grouped according to function in the testbed. The description is of the planned Testbed, not just those portions that were operational at the end of FY89. More detail on individual components can be found in Appendix B.

### 3.2.1 Basic Communications

The Testbed uses two levels of multiplexing. At the low-speed level we have the military AN/FCC-100 multiplexer which produces composite signals (as described below) that are applied as inputs to the higher-level multiplexers, namely Intraplex TDM-153 Channel Banks. The latter are standard commercial channel banks that are functionally equivalent, in all respects that affect MITEC, to the military AN/FCC-98 multiplexers: they combine DS0 signals into 1.544 mbps DS1 bit streams that are carried between the TCFs on twisted pair wires. Channel cards are available to support 56 kbps synchronous data and 64 kbps PCM voice channels as well as a multiplexed channel capable of handling up to 5 low-speed synchronous data users concurrently.

The TDM-153s do not offer any means for control or direct monitoring by MITEC. They are configured by the type of card plugged into each slot and by the settings of switches on the cards. Alarm conditions are indicated by relay closures that are sensed by MITEC through the Datalok 10A device described below. The sensed conditions are transmit failure, loss of received frame/signal, and power failure. A fourth alarm is generated to indicate a problem at the remote end, but we do not plan to use this alarm in MITEC.

At the lower level of multiplexing, we have FCC-100 Low Speed Time Division Multiplexers. These LSTDMs are military units loaned to us by DCEC in support of the development effort. FCC-100s have RS-232 control connections that allow dynamic reconfiguration of both individual channels and the aggregate data rate. They support a maximum of 16 channels and an aggregate rate up to 512 kbps. We have only enough port cards to support two synchronous and two asynchronous data channels of the possible 16. These channels are normally configured to run at 2400 bps, a rate that matches the modems used in the tail circuits.

The FCC-100 aggregate normally runs at 56 kbps and is carried by the 56 kbps T1 channel, but it can be configured to run at 9600 bps, and, still carrying three of the 2400 bps channels, be sent through a 9600 bps 4-wire modem and thence either through the VF T1 channel or the 4-wire phone line to restore service for some of its channel users. The FCC-100 provides relay closures that can be sensed by the Datalok 10A (see below) to indicate power failure, loss of received aggregate frame, three loopback states, and a

fault detected by the built-in test equipment. These conditions (except power failure) and others of interest can also be sensed via the RS-232 control port.

The testbed includes two different kinds of modems. Hayes "Smartmodem 9600" modems are used on the 4-wire trunk line between the TCFs. Codex Model 2510 modems are used on the 4-wire tail circuits supporting the 2400 bps data users. For a time, 1200 bps 2-wire modems were a part of the Testbed plan, but we were advised by our steering group of tech controllers that 2-wire data communications were of little interest in the military, so we removed them. The Testbed includes spare Codex 2510 modems so that restoration of a tail circuit can be demonstrated when diagnosis indicates that the problem is in the modem at the TCF end of the tail circuit. No spares are available for the 9600 bps modems on the 4-wire trunk.

None of the modems can be controlled or monitored by MITEC. The Codex modems have the capability to be remotely monitored and programmed but only through a proprietary network control system, and the use of such a system would violate our design principle of using only public protocols. Consequently, MITEC must ask its operator to enter any status information from the modem indicators needed during a diagnosis involving the modem. Similarly, MITEC must ask the operator to carry out any needed configuration changes.

The T1 and phone lines in the Testbed are realized by twisted-pair wires. The lines have attenuators wired in to match the attenuation found in real telephone circuits. There can be as many spares as desired. We do not propose to provide spare lines for the tail circuits since such are not usually available in the real TCF world.

Figure 3.2 shows the basic communication equipment and circuits in the Testbed. For simplicity, the figure omits all patching and test point access elements. We have a total of seven 2400 bps modems in the Testbed, and the figure shows them all at one TCF, a configuration that we favor over splitting them between the two TCFs because it offers us the greatest flexibility in carrying out experiments. For example, the components of the two synchronous tail circuits at the WEST TCF can be reconfigured into a single circuit with two tails instead of being used in the two circuits shown. Such a circuit is typical of "on-base" circuits of interest in real world tech control situations.

### 3.2.2 Patching and Circuit Accessing

To perform its basic mission of fault isolation and circuit restoration, MITEC needs to be able to access circuits at appropriate places for test purposes and to patch around failed segments or devices. To the extent that this mission is to be

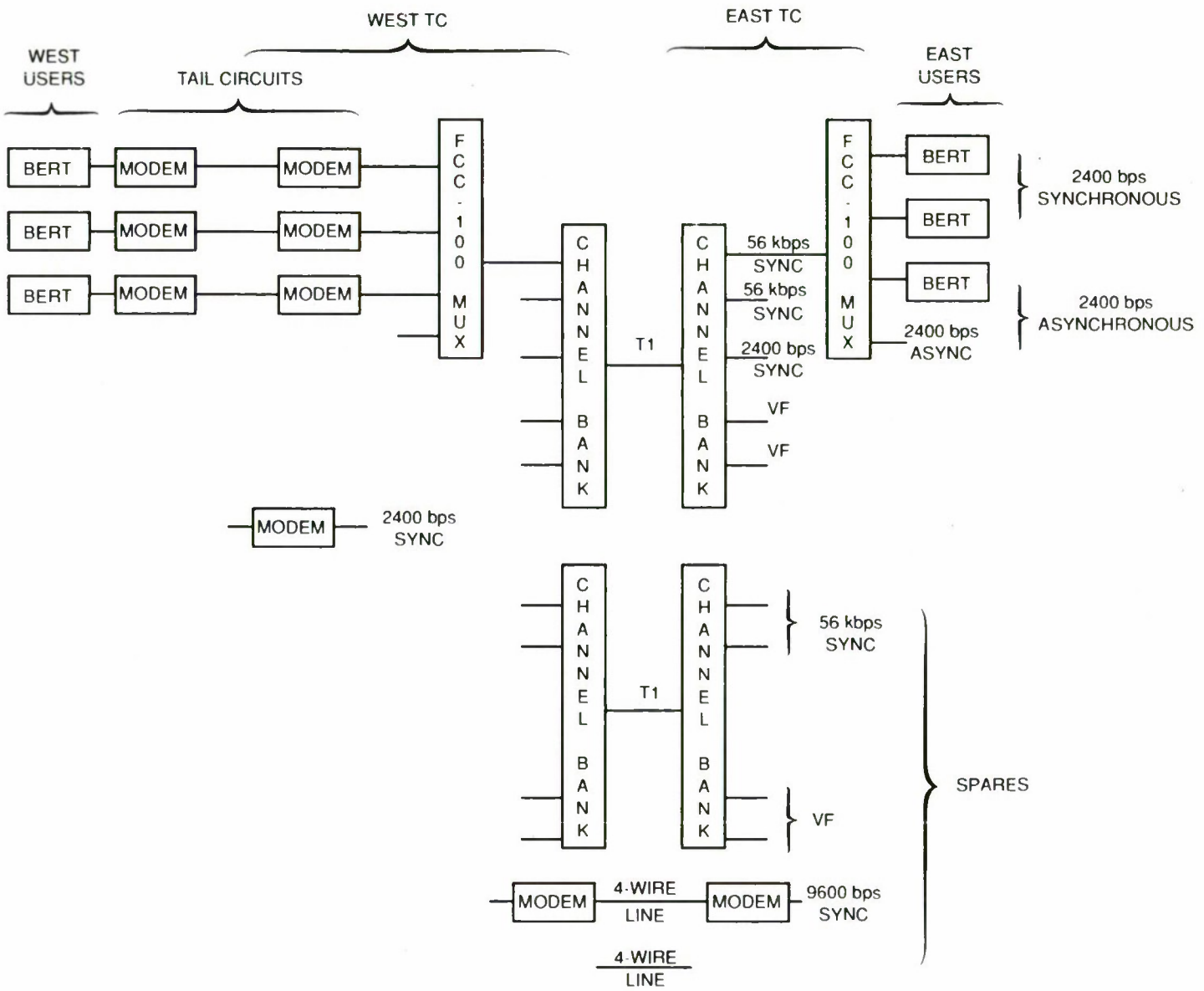



Figure 3.2
Testbed Communications Equipment
and Circuits

performed automatically, MITEC must be able to carry out these functions electronically. An electronic matrix switch supports both functions, and we have included such a switch in each TCF in the Testbed. It provides test access at every point where a patch can be made, and that access is free (no additional cost for the TCF) if the patching capability is required. However, there are many points in a TCF where test access is needed but patching is an infrequent event, and the cost of access via a switch may not be justified. To demonstrate MITEC's performance in such situations, we have included conventional manual patch panels in the Testbed, as well as a test access switch that allows MITEC to access circuit points for measurement purposes but does not support patching. The latter equipment provides access at a lower cost per test point than does a complete electronic switch as well as offering true metallic connection to the circuit as opposed to the digital sampling used in the switch. The manual patch panel further reduces the cost, but introduces an "air gap" (a human must be asked to carry out the operation) with consequent slower speed and increased probability of error.

For the matrix switch we chose the Telenex Mini-Matrix Switch (MMS) through the Laboratory's regular competitive procurement procedure. The MMS won the competition on a cost basis because its general architecture was well matched to the small scale of our testbed. This particular Telenex product would not be applicable in a larger size TCF, but Telenex and other manufacturers offer larger switches that would be appropriate for such use.

In the Testbed, the MMS switches RS-232 and RS-449 data circuits and 4-wire VF circuits. It also provides for passive monitoring and the insertion of test equipment. The Testbed plan calls for three monitor/test positions; one to monitor RS-232 signals through a digital storage oscilloscope, a second to monitor and test VF circuits using a communications test instrument, and a third to connect a protocol analyzer for loopback and bit error rate tests (see below for descriptions of these test instruments). At the end of FY89, only the first two monitor/test ports were in use.

The RS-232 monitor can be used to examine RS-449 circuits, but the signals appear as unbalanced RS-232 signals instead of the balanced form in the RS-449 circuit. Also it should be noted that the MMS sampling technique for NRZ data signals uses only one bit per sample so that mark and space voltages are standardized by passage through the switch. Since the MMS monitor port sees the signal inside the switch after standardization, it cannot be used to diagnose a weak or marginal NRZ signal problem.
For VF signals the MMS uses PCM encoding, and care must be taken to set transmit and receive signal levels so that quantization effects do not introduce unnecessary distortion and noise into the signals passing through the switch and being measured at the monitor port.

In the testbed, further test access is provided by a Hekimian Model 3200 Test Access Switch (Hk-3200) in each TCF. One of these systems was purchased for the testbed. The other is on loan from AFCC. The Hk-3200 is a system designed to provide bridge and split access to 2, 4, 6, and 8-wire circuits in telephone system applications. Relays are used to implement the access switch. These are grouped in units called "shelves". A shelf can hold enough relay cards to support 50 8-wire circuit test points. In the testbed we have two shelves in each TCF and enough relay cards to put test point accesses at all interesting points in the circuits.

Each Hk-3200 has two test positions controlled by independent RS-232 lines from MITEC. Since we want to have Hk-3200 test points in both data and VF circuits, we plan to feed the access line outputs from one test position to a digital oscilloscope and the lines from the other to a communications test instrument (see below for further description of the test instruments). Each test position can independently access any test point on either of the two shelves, but a shelf can support only one access at a time with the consequence that an attempt to access a test point will be blocked if the shelf is busy. To minimize such blocking we are putting test points for VF circuits on one shelf and those for digital circuits on the other.

Since the testbed has two switches that can provide circuit access but only one set of test instruments, we need either to add an additional switch to connect the test equipment to the desired access switch or to subordinate one switch to the other. We have chosen the latter course. Prior to the end of FY89, the Hk-3200 test points were not completely wired into the circuits, and the test instruments were directly connected to the monitor ports on the MMS. In FY90, in order to make the best use of both switches, we propose to connect the test instruments directly to the Hk-3200 test outputs and wire the MMS monitor port outputs to Hk-3200 input test points. Because there are ten leads of interest in some circuits and the Hk-3200 can access only eight at a time, we plan to connect the MMS monitor port to two 8-wire Hk-3200 test points. Between the two, all interesting combinations of the ten signals can be presented to the digital storage scope for analysis.

In the new configuration, a test access using the MMS will require the Hk-3200 to select the MMS monitor port and the MMS to monitor the desired signal. This subordination choice allows the use of the Hk-3200 relays to get linear access to NRZ signals whenever an appropriate test point is available, but it exposes the measurement process as well as the NRZ circuits themselves to a possible source of noise and interference by passing the signal to be measured through the considerable length of wiring associated with the Hk-3200. The Hk-3200 is designed to be wired using 25-pair telephone cables which provide little protection from cross-talk between NRZ signals which have frequency components well outside

the VF range. Preliminary experiments with a partially wired Hk-3200 shelf showed some cross-talk but not enough to cause impairment. More thorough experiments will be carried out when the wiring is complete.

The testbed design does not provide any means for MITEC to electronically access the DS1 bit streams carrying the T1 trunks between the TCFs. We have an instrument in one TCF that can analyze DS1 signals, but a manual patching action will be required to connect it to the signal in question.

In order to allow MITEC to test tail circuits, the testbed includes Model R-4 Programmable Responders on each such circuit. These devices, built by Domain Systems, Inc. are connected between the 4-wire lines and the modems at the user ends of the tail circuits. They can be commanded by DTMF tone bursts from the TCF to disconnect the remote modem, apply a quiet termination, loop the line back to the TCF, or send back a 1004 Hz test tone. In a real TCF, these devices minimize the need to involve user site personnel in testing tail circuits and/or to send TCF personnel to user sites. In the testbed, MITEC can use this capability in conjunction with its VF test instrument in both fault isolation and quality control (QC) testing.

Another planned testbed element is an AT&T DACS-II Digital Access and Cross-Connect System. This system provides switching and access to T1 channels. It is included in the Testbed because it is a component of the Digital Patching and Access System (DPAS) currently being deployed for military use. The DACS-IIs in DPAS will be controlled by a network of computers that will use appropriate algorithms for rerouting circuits and trunks to improve the survivability and efficiency of the military transmission network. In the testbed, we will have one DACS-II when it is installed in FY90.

We do not plan to connect the DACS-II directly to either MITEC, but instead to a separate computer that will emulate the functionality of the DPAS. The emulation will split the one real DACS-II into two virtual units, one each to be associated with each TCF in the testbed. We will try to adhere to industry standard protocols for network management as and when they become available. We will pass the T1 trunks between the TCFs through the DACS-II and connect RS-232 order-wires between each MITEC and the DPAS emulation computer. Figure 3.3 shows this configuration schematically. The order wires will carry requests from MITEC for the rerouting of circuits by DPAS and their corresponding responses as well as reports resulting from any alarm conditions detected by the DACS-II as it monitors the T1 trunks. It should be noted that the T1 lines shown going between the EAST and WEST DPAS may be either real or emulated. There would be no differences as far as the MITECs are concerned.

### 3.2.3 Alarm Sensing

Alarm sensing in each TCF is provided by Datalok 10A/MICRO Remote Stations manufactured by the Pulsecom Division of Hubbell Inc. These are microprocessor-based monitoring and control units that, when polled, report the status of binary or analog inputs and send control outputs that set the state of relays in the unit. Dataloks are used in the TRAMCON system in Europe to monitor and control microwave equipment at remote sites. In such an application, the polling feature allows many units to share a single communication link. In the testbed there is only one unit in each TCF, and polling is not needed to separate device responses, but it is used to allow MITEC to control the times at which alarm conditions will be processed.

The binary inputs to the Dataloks are used to sense alarm conditions in other testbed equipment. Of three alarm reporting modes offered by the unit, we use the "normal" mode which identifies each input transition as a change of state and remembers the sequence of changes for reporting when polled. Because the state changes are queued between polls, MITEC must poll as many times as necessary to get a response indicating that no more state changes are queued. It then uses the last reported change to figure out the current state.

The only alarms sensed in the testbed are those generated by the FCC-100 and TDM-153 multiplexers (see Sec. 3.1.1). Alarms from other devices such as the modems do not generate signals that can be sensed by the Datalok 10A.

The control output feature of the Datalok 10A is used to switch the probe inputs of the digital storage oscilloscope among the signals of interest at the test point being monitored (see below).

### 3.2.4 Signal Measurement

For the measurement of VF signals, the testbed has a Hekimian Laboratories, Inc. 3700 series Communications Test System in each TCF. The systems are not identical. One is a Model 3705, the other a Model 3703. The model number only partially defines the system's capabilities because a variety of options are available to enhance a basic VF circuit testing capability. The model number 3705 indicates a capability to analyze PCM circuits. The number 3703 indicates a system designed for unattended operation (no front panel controls or indicators). Both systems can make signal level measurements, generate test signals, measure frequency responses and signal-to-noise ratios. They can generate the DTMF tone sequences needed to command the programmable responders on the tail circuits. The systems can be programmed to make sequences of measurements and to cooperate with each other using a signaling path through the circuit being measured, but we do not expect to be using these features in conjunction with MITEC.

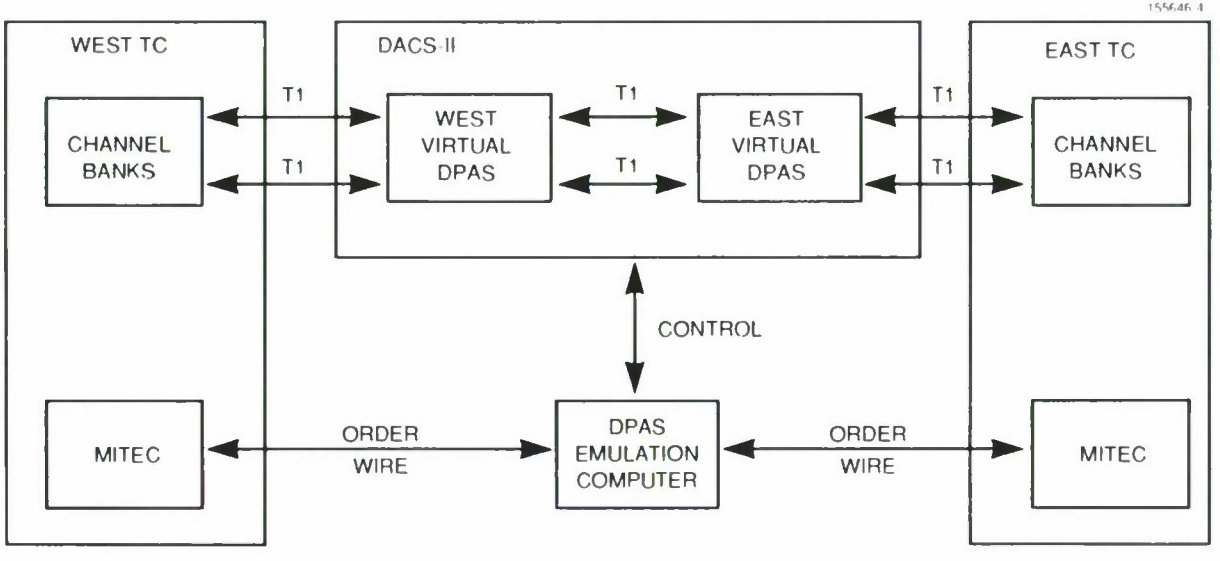



Figure 3.3
DPAS/DACS II in the Testbed

The Model 3703 has an option installed allowing it to measure return loss in 2- and 4-wire circuits and to make peak-to-average ratio measurements designed to show the effects of envelope delay distortion, amplitude distortion, and poor return loss on voice band data signals. The Model 3705 has the capability to analyze the DS1 signals produced by the channel banks. In addition to assessing the overall quality of the DS1 signal, the 3705 can pull out individual PCM channels from the DS1 bitstream for testing purposes. This feature will allow MITEC to do QC testing of the VF channel cards in the channel banks.

Because the VF test systems are not identical, MITEC must have database entries indicating the capabilities available in each TCF, and we must take care in generating demonstration scenarios to have problems occur in the right place so that the appropriate equipment can be used in the diagnosis. In a real-world application of MITEC, it would be very desirable to have matching equipment capabilities at all sites. Also, additional system options would be needed to support other tests required by DCA Circular 310-70-1 such as impulse noise, envelope delay, and phase jitter.

For measurement of data signals, the testbed has one Philips PM3352 Digital Storage Oscilloscope in each TCF. These scopes concurrently sample two input waveforms at rates up to 100 million samples per second and store the samples for retrieval and analysis by MITEC. The samples have 8 bits of amplitude information, and under typical conditions, each waveform is represented by a set of 512 sample points. All the usual control settings for oscilloscope usage such as sweep rate, amplitude sensitivity, and triggering mode can be controlled by MITEC through its RS-232 connection to the scope.

Because the scope can sample only two waveforms concurrently, and there may be as many as five (10 for balanced signals) of interest in a synchronous data circuit (two data signals and three clocks), it was necessary to provide a means of selecting the leads to be sampled. We chose to use the command outputs provided by the Datalok 10A/MICRO rather than install an additional device in the Testbed for this purpose. The command outputs drive a set of ten latching relays that are wired to present all useful pairwise combinations of the ten signal leads to the two scope inputs. These combinations allow clock-to-data signal timing relationships to be observed as well as both halves of all balanced signals. Clock-to-clock timing can also be measured allowing MITEC to diagnosis a class of timing problems, that while rare in TCF experience, can be difficult to detect by tech controllers.

There are other leads in the RS-232 connectors that cannot be seen due to limitations in the present signal selection arrangement. An example is the Clear-to-Send from a modem. In a real-world application of MITEC a more general selection capability should be included so that all signal leads could be examined. We would also

recommend that a scope capable of simultaneously measuring 4 waveforms be provided.

Since the Hk-3200 provides access to VF as well as digital signals, the scope can be used to analyze the VF signals produced by modems if programs are written to compute the spectra of such signals and to classify them appropriately. The question of interest is to determine whether or not modulation is present in the signal. Tech controllers listen to the VF signals to make such a determination. As of the end of FY89, MITEC did not have the software or hardware (Hk-3200) capability to analyze modem waveforms, but it has been suggested that it would be desirable to add such when the necessary resources become available.

Another instrument needed in the testbed is a device capable of generating test data signals and measuring bit error rates. Without such an instrument MITEC could not perform the loop testing essential to complete fault isolation, data circuit QC and spare testing. We have chosen the Digilog Model 620i Protocol Analyzer for this purpose. Of the instruments evaluated, the Digilog family offered the most complete control of analyzer functionality from its RS-232 control port, and the Model 620 matched our speed and capacity requirements and budget limitations. The "i" option on the Model 620i gives us a capability to interface to ISDN networks, a likely future need in tech control applications. The instrument has many capabilities for monitoring data communications and for interacting on a protocol level with other entities in data communication networks. We will use only a fraction of these in our initial MITEC implementation to support bit error rate and loop testing. Two analyzers, one for each TCF, were delivered at the end of FY89, and plans call for their installation in the Testbed as soon as work on the Ft. Detrick/Pentagon demonstration is completed.

### 3.2.5 Fault Insertion, Line Degradation

In order to test MITEC's ability to diagnose problems, it is necessary to introduce real problems into the testbed circuits. Toward this end we have provided a Fault Insertion panel in each TCF and devices for modifying the quality of the communication lines. To introduce a fault into a circuit, we manually patch the circuit through the panel, and open switches on the panel to interrupt the desired circuit leads, e.g., received data and/or clock. The panel has enough connectors and switches to concurrently introduce faults into four RS-232 and one RS-449 circuits. There is a switch for each lead in the circuit so that leads can be independently interrupted. By patching the panel into circuits at appropriate places, we can produce symptoms that look like device or module failures in so far as their overall effect is concerned. Of course, if one examines the signal in detail at the point of the fault, one sees an open circuit, which is not the same as what one would see if there had been a real module failure.

If we had defective cards and/or devices, we could use them for testing, but the procedure would be much more cumbersome than simply changing a switch on our panel.

Faults useful for testing diagnoses can also be introduced by unplugging cards, inserting patch cords into manual patch panels, and turning off the power to equipment such as modems that do not generate power-off alarms.

Another mechanism for introducing problems for MITEC to diagnose is to manually change the configuration of some configurable device. For example, the FCC-100 multiplexer has front panel buttons that can be used to set its configuration. Changes introduced by button pushing may or may not produce alarm events sensible by MITEC, but in any event, it is possible to cause problems in the circuits ranging from dramatic, e.g., complete loss of communication due to mismatch of the two ends, to subtle, e.g., occasional bit errors due to incorrect timing. Similarly, manual changes can be introduced in the modems and the user BERTs to cause dramatic or subtle problems. In the case of the FCC-100, MITEC can discover the change and correct it automatically, but manual changes elsewhere, for example in modem programming, while often diagnosable, require manual action for confirmation and correction.

The testbed has two telephone line simulators that can be used to introduce problems into the lines between the TCFs and/or the tail circuits. These are Processing Telecom Technologies, Inc. Model PTT 5100 Telephone Line Simulators. One of them will be augmented in FY90 with a PTT 5151 Echo/Advanced Impairments Simulator. The basic units can simulate both 2- and 4-wire lines. In the 4-wire situation the simulation takes place in one direction only. The return wire pair is passed through the unit without any impairment. In the testbed the simulators can be manually patched to cause problems in either direction on the lines. A number of standard line types are built in, and arbitrary types may be created by specifying high and low band amplitudes and envelope delay distortions. With any type, attenuation and signal-to-noise ratio may be specified. The 5151 adds the capability to introduce echos, linear and non-linear distortion, frequency offsets, phase hits, jitter, gain hits, dropouts, impulse noise, interferences, and satellite delays. In addition to introducing problems in the phone lines for diagnosis tests of MITEC, these units can produce realistic data for QC testing of the lines.

To introduce problems in the T1 lines, the testbed has a single Wandel & Goltermann PKN-1 Line Simulator. This unit is basically a calibrated attenuator that operates in one direction on the line. At high attenuation levels the channel banks lose framing producing an outage of the T1 trunk. At attenuations within a few dBs of the failure point, bit errors occur that can be sensed by the Hk-3705 test instrument and seen by the user BERTs. We assume that the DACS-II will also report the degradation in signal quality

associated with this attenuation. While it would be desirable to be able to introduce noise without attenuating the signal because such situations can occur in real circuits, we do not believe that it will be necessary to create those cases to validate MITECs ability to diagnose such problems.

### 3.3 Testbed Diagram

Fig. 3.4 shows a diagram of the WEST TCF in the testbed. It represents a goal configuration toward which we will be working in FY90. The actual configuration at the end of FY89 had no DACS-II, no Hk-3200 test access points, only one T1 multiplexer, and just two user circuits, one synchronous and one asynchronous. A corresponding diagram for the EAST TCF would be the same as far as the multiplexing is concerned but would have no modem tail circuits, i.e., the user BERTs would be directly connected to the DTE boxes on the FCC-100 side of the center. The figure shows the communication equipment and the points at which patching and access can be directly achieved by MITEC without human assistance. Manual patch panels, test equipment, and devices for introducing faults and/or signal degradation have been eliminated to simplify the figure.

The boxes in the columns labeled "MMS" show the interface elements and possible connections that are internal to the matrix switch. The interface elements for NRZ data signals are labeled "DTE" and "DCE" according to the kinds of ports to which they are connected, e.g., DCE elements are connected to modems. The elements for VF signals are labeled "CM" and "EM" (for Channel Module and Equipment Module, respectively).

The solid lines show the normal flow of the circuits and trunks. The "null" label next to a line indicates a crossover between transmit and receive wires in the cable that is so marked. The term "null" derives from the computer field where it is applied to a cable called a "null-modem" that allows a pair of like devices, such as terminals, to inter-communicate.

The dashed lines show possible patches that can be made to restore service. It should be noted that while the MMS can make a connection between any DTE and any DCE (also any CM and EM), it cannot connect a DTE to another DTE. Thus a patch that would directly connect the tail circuit VF line from the point at which it enters the TCF to a VF channel on a channel bank cannot be carried out because it would require two CMs to be connected. If such patching were desired, we would need to provide a pair of EMs connected with a null line such as the diagram shows for the NRZ circuits. This configuration at the "center" of the TCF allows full patching flexibility in both directions.

The isolated CM and DTE boxes with looped lines indicate possibilities for MITEC to introduce loops for testing purposes.

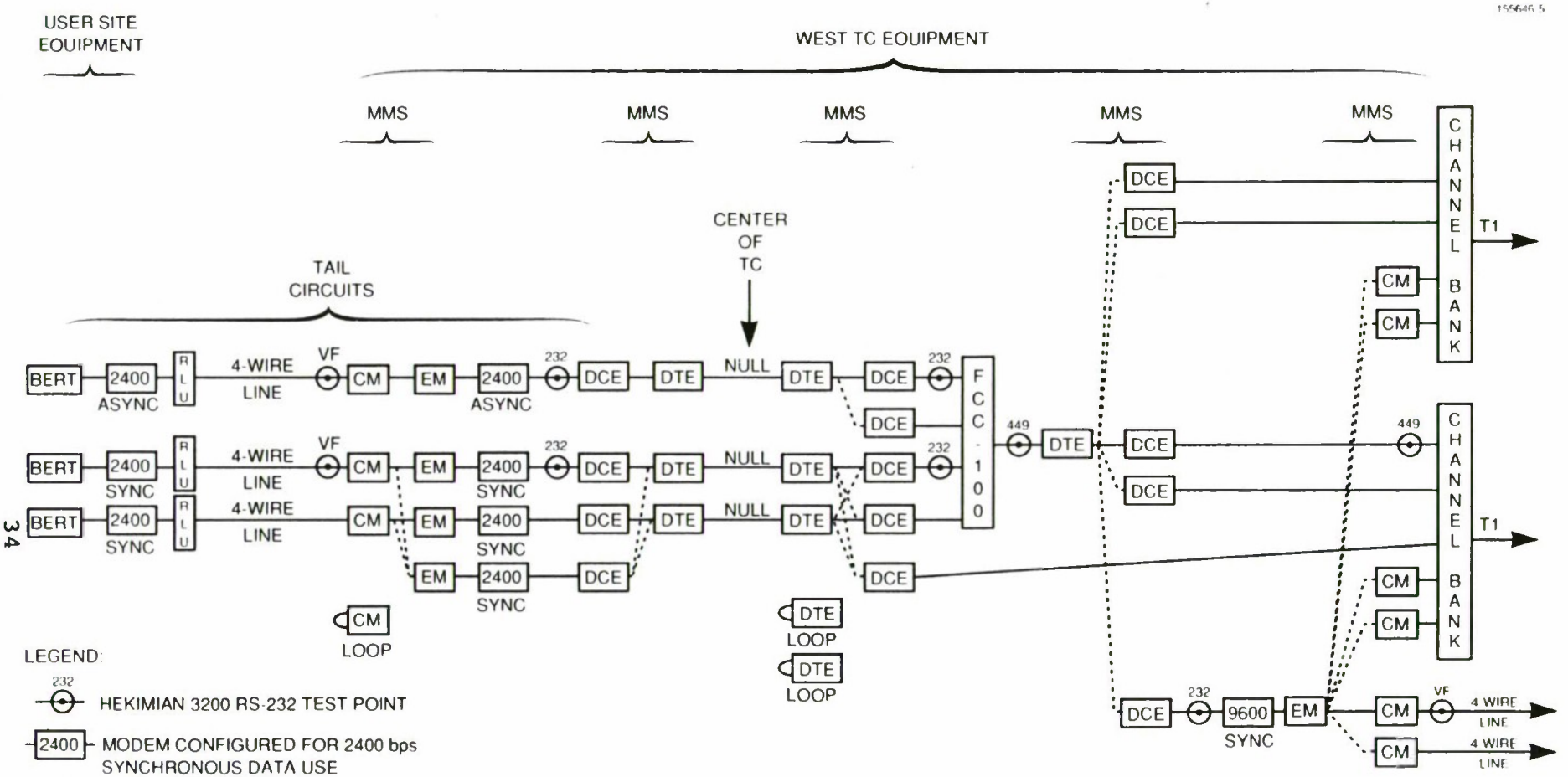



Figure 3.4
West TC Diagram

The CM loops have attenuation to match that of a nominal phone line (16 dB).

The concept of the "center" with its associated null line allows the signal in the receive sense from one direction to become a transmit signal in the other. We do not use a null center for user circuits that do not involve modem tails since there are no patching possibilities in the user (BERT) direction. There is a cost in providing an additional DTE interface, and we believe that in a real TCF application, it is unlikely that the interface would be provided for the sake of generality. Its absence, however, poses an additional complexity for the MITEC software.

In the figure the small labeled circles indicate the locations of test access points provided by the Hk-3200 test access switch. We show only a few test access points in the figure, but we may well add more to allow a full diagnosis without any use of the MMS monitor capability. At a minimum we want to be able to make measurements at least one place where each different kind of signal appears. We also want to be able to use the Hk-3200 test access points to diagnosis faults in the MMS interface units.
The labels next to the circuits indicate the type of signal present at the test point. There are two kinds of NRZ signals shown, RS-232 and RS-449. We use this terminology because it corresponds to the type of interface unit to the MMS that handles the signal at the point at which it is labeled. Actually, the signals at some of the test points are mixtures of MIL-188 and RS-232 or RS-449 since the FCC-100 multiplexers generate MIL-188 signals. The MMS interface units will operate satisfactorily with those signals and the FCC-100s are happy to receive RS-232 signals when properly configured. However, MITEC must know that the mixed signals are present at the test points so that its signal quality assessments will be correct. It does this by associating the signal type with the device that generates each signal rather than with the test point which is always between devices.

# 4.0 MITEC System Organization

## 4.1 System Architecture

### 4.1.1 Symbolics Computer

The Symbolics 3600 family of computers was selected as the platform on which to develop the MITEC system. The reasons for this choice can be summarized as follows:

1. The precursor ETC project, Expert Tech Controller, was done on the Symbolics computers. This provided a base of knowledge, experience, and software that could be carried over to MITEC.

2. The Symbolics Genera operating system provides an efficient, highly productive environment on which to develop software. The LISP programming language combined with the Flavors object-oriented programming system is well suited for the symbolic processing needed in a MITEC system. The entire environment is especially appropriate for the fast prototyping style of development we have used to produce MITEC.

3. Superb graphics and interactive facilities are available, including a screen resolution of 1280 by 795 pixels, a three-button mouse, a wide variety of type fonts, windows and panes, and several different styles of menus.

The Symbolics computers serve as the main computers for the development and execution of MITEC. Since all of the Symbolics computers at Lincoln Laboratory are linked together on an Ethernet, it has been possible to designate one computer as the repository of the MITEC source files. Each software developer has full-time access to one of the networked Symbolics computers.

The MITEC system is structured so that one can "create" one or more MITECs on a given Symbolics computer. These MITECs can then operate independently of each other and are logically unaware of the location of their neighbor MITECs. This flexibility enables the developers to test from one Symbolics computer the MITEC to MITEC communication mechanisms.

MITEC is written entirely in Zetalisp combined with the Flavors object-oriented system. Its size is approximately 20,000 to 25,000 lines of code.

### 4.1.2 Sun Computer

MITEC has a need to communicate with the various equipment items in the (simulated) TCF and with other MITECs. A Symbolics computer has only two serial ports and its system software is not well suited for real-time communication. It was therefore necessary to

provide a front-end processor which could support real-time communication with a variety of devices. We chose a Sun workstation, equipped with an ALM-2 16-port serial line multiplexer, as this front-end processor. The Sun computer provides the low-level communication capabilities and has minimal TCF-specific knowledge. Such knowledge and the complexity of device-specific protocols reside in the SYMBOLICS computers. For uniformity, all MITEC communication to devices as well as to other MITECs goes through the Sun computer.

As the communication software on the Sun we began with the existing program TODOWN, which provides communication with multiple down line devices via RS232 connections. Numerous modifications have been made, as described below.

The Command-Response Paradigm: In previous uses, TODOWN served as a means for a person to sit at a terminal of a host computer and communicate with a number of downline computers as if he had a terminal on each one of them. In the MITEC context the person is out of the loop and it is the Symbolics computers themselves which do the communicating with the equipment. We therefore developed a master-slave command-response mode in TODOWN. In this mode the Symbolics computer is considered the master and can thereby issue commands. After checking the command for validity, TODOWN sends the command to the specified downline, the slave. A subsequent communication from the slave is considered to be a response to the command and is then returned to the most recently commanding master. The termination of a response is indicated by the appearance of a user-specifiable prompt request from a computer for more input. A one-command-at-a-time discipline is imposed such that an attempt to send a command to a device when another command to the same device is outstanding results in a "BUSY" response.

In general, several such command-response dialogs may proceed simultaneously between processes in the Symbolics and devices in the TCF. To provide a means for associating a response from a device with the antecedent message, a message-id field is included in the original request. This message-id, chosen by the master, is remembered by TODOWN and is later returned to the master as part of the response from the device.

Some devices have subcommands which provide different prompts from those provided when expecting a top-level command. To handle such devices, TODOWN provides a means for dynamically changing the prompt that is serving as the response terminator. In some situations it is necessary to use a count of characters, rather than a prompt, as the terminator of the response. TODOWN provides a command for specifying and changing this count. The arrival of characters from a device in the absence of an outstanding command results in a notification to MITEC which can then "flush" the buffer and read the characters.

Communication with Other MITECs: Such communication exchanges are regarded as master to master communications, and do not involve waiting for a response. TODOWN makes a distinction between a local master and a remote master. A local master is a MITEC running on a Symbolics that is directly attached to the serial-port board on the Sun. On the other hand, a remote master is a MITEC that is attached to another Sun and therefore can be reached from the local Sun by sending a message to the TODOWN running on the remote Sun with directions to pass it on to the Symbolics computer. A remote master situation occurs when one has two MITECs, each running on its own Symbolics and Sun, with a phone line connecting the two Suns to each other.

For development purposes we may at times wish to run two MITECs in one Symbolics computer. From the point of view of either system, the identity of the computer on which the other MITEC is running should be of no concern. We therefore introduced a "map" command which enables MITECs to communicate with each other by using logical names; TODOWN directs the messages for a given MITEC to the computer on which it is running.

Miscellaneous Features: TODOWN itself is considered a device which can be commanded; this enables MITEC to send query and other commands to TODOWN and then to analyze the responses to determine current status of the devices.

An optional logging facility is available in TODOWN; when it is enabled, all commands and responses are logged on TODOWN's terminal.

An automated TODOWN startup mechanism was implemented. By typing the name of the startup file to the UNIX Shell running on the Sun one invokes TODOWN and issues the appropriate TODOWN commands to attach the various devices needed by MITEC and to "boot" each such device. The booting process attempts to place the device in an appropriate mode for subsequent use by MITEC.

### 4.1.3 Communication with Devices

In this section we describe the devices with which MITEC communicates. The descriptions are oriented from the point of view of the characteristics of such communication, the peculiarities we encountered, and the provisions we had to make in TODOWN and elsewhere in order to accommodate these peculiarities.

The extensions to TODOWN to support the various devices in MITEC have prompted the generation of a document outlining the device communication characteristics that are desirable in a device that is to be included in MITEC. This document is included in appendix B.

#### 4.1.3.1 Philips Digital Storage Oscilloscope PM3352

This device does not present feedback when one communicates with it. It does not echo the input it receives, does not prompt for the next command, and does not generate error messages. Our usual mode of communicating with such a device is to use commands which elicit responses, even though we are not interested in the contents of the responses. In this way, we can establish that we are successfully communicating. We also use this mechanism to force a prompt from the scope for another command.

The scope's speed and other modes are not remembered across power outages. We therefore had to generate some specialized code in order to initialize the scope. Another major problem we faced with the scope was attempting to maximize bandwidth by adjusting the scope transmission rate. After considerable experimentation we finally settled on the following approach. We use two lines between the Sun and the scope: the Sun to scope line at 1200 baud and the scope to Sun line at 2400 baud. Since XON-XOFF is inoperative in a two-line mode, we use a binary output mode on the scope for waveforms. (In the binary mode each point in the waveform is sent as an 8-bit value rather than as ASCII characters that represent the value. Therefore, the 512 points occupy 512 bytes rather than about 4 times as much if the points would be sent as 3-digit ASCII integers with separating characters.) This solution is somewhat fragile in that values could be lost due to the lack of flow control. Another disadvantage of this approach is the slowness due to the low scope output speed (2400 baud), but running faster dramatically increases the risk of loss of points in the waveform.

#### 4.1.3.2 Datalok 10A with Relay Scanner

This device communicates not in ASCII but in binary values. This makes it difficult to experiment with the device. To solve this problem we created a "virtual" octal device in TODOWN, which transmits and receives in ASCII the representations of octal integers in the range 0 through 377. One can send to this device a string of such octal integers and receive a response as octal integers. TODOWN provides the service of converting between the octal integers and the binary bytes that the device requires. This mechanism has worked quite well.

#### 4.1.3.3 TELENEX Mini-Matrix Switch

The initial version of the TELENEX mini-matrix switch was designed for operation from a dedicated terminal. As such, it formatted its output for the terminal and relied upon the human operator to determine the current status and what to do next. Commands were menu driven, descriptions of commands were provided, output contained screen formatting characters, and there was no XON-XOFF flow control support. The lack of support for XON-XOFF combined

with the large outputs for certain commands would have made it difficult, if not impossible, to incorporate the switch into MITEC, since there would be a high probability of lost output from the switch. Furthermore, there was no prompt at the completion of a response from the switch.

In order to make output from the switch more readable a new-line emulation mode for a VT-52 terminal (one of the terminals supported by TELENEX) was introduced in TODOWN. In this mode, TODOWN scans incoming characters from the TELENEX searching for VT-52 new-line formatting sequences; these are then converted to the ASCII new-line sequence to make it easier for the MITEC developers to read and understand the switch responses.

The current version of the mini-matrix switch provides a much closer fit to the needs of MITEC. There is a dip-switch in the device which permits selection of operating mode: old style for human interaction and a new style intended for computer interaction. In the latter mode, which we use in MITEC, menus and error messages are suppressed, XON-XOFF flow control is supported, and control-V can be used to place the device in a known state. A prompt (a single character) is provided after each character is received by the switch.

Our current style of using the mini-matrix switch involves a set of subroutines that we produced for performing standard switch actions. For example, one subroutine accepts a DCE and DTE and commands the switch to connect one to the other. Internally these subroutines operate in two stages. First, they change the prompt expected by TODOWN to be the sequence of characters that the switch returns in response to the characters sent to it to produce the desired action. Then, they send off the characters that are in effect the command.

Obtaining a DCE-to-DTE (or vice versa) dump of the switch's memory produces a large amount of output because of the formatting into columns for convenient human readability. We have produced subroutines which process this output in order to extract the required information.

One major deficiency of the switch is that placing one of its ports into loopback mode can only be done manually. It is not possible to change or detect the loopback status from the computer. This prevents certain diagnostic tests from being carried out.

#### 4.1.3.4 AN/FCC-100 Multiplexer

The FCC-100 has a command structure which contains subcommands. When the device is expecting a subcommand it issues a prompt that is different from the prompt that it issues when it is awaiting a top-level command. Since TODOWN's master-slave communication with a device relies upon prompts as signifying the end of a response

from a device we need to change TODOWN's record of the prompt whenever we enter and exit a subcommand.

#### 4.1.3.5 Hekimian 3701, 3703, 3705, 3200

These devices seem quite well-behaved for communication from a computer. The only problem encountered is the appearance of a herald or a prompt from the device when we "open" a connection to it from the SUN. This herald/prompt is treated by TODOWN as a response without a preceding command and results in a superfluous notification to MITEC about it.

## 4.2 SCHEDULING SYSTEM

### 4.2.1 SCHEDULING OF RESOURCES

This section describes the scheduling of two types of resources in MITEC: processes that execute as part of MITEC and access to test/measurement devices.

#### 4.2.1.1 PROCESS SCHEDULING

MITEC consists of a number of processes that execute asynchronously. Since the Symbolics computer contains only one processor, the MITEC processes must execute sequentially rather than simultaneously. The scheduling of such execution must take two considerations into account. One is that each process gets a reasonable amount of computer time during which to execute. The other is a requirement that inter-process dependencies be honored.

The requirement that each process get its fair share of execution time is left to the Genera operating system on the Symbolics computer. Its internal scheduler attempts to apportion execution time in a reasonable manner subject to many considerations. One such consideration is process priority: higher priority processes execute whenever possible in preference to lower priority ones. We make use of process priority by giving the MITEC Scheduler process higher priority than other MITEC processes. (The MITEC Scheduler is described further in the next section.)

Inter-process dependencies occur when one process depends upon the completion of some activity by another process before it can meaningfully continue running. For example, a diagnosis process that has requested a device to perform a measurement cannot continue until it obtains the results of the measurement. We implement such dependencies by means of the keyboard input buffers associated with each process. Using these buffers, processes send messages to each other requesting actions and indicating completion of actions. When process A (PA) needs an action performed by process B (PB) it sends a message to PB to perform the action. PA then attempts to read from its input buffer, finds it empty, and hangs. Meanwhile, PB reads its buffer where it finds the request.

Eventually, PB completes the action and sends a message to PA indicating completion and the results, if any. PA's buffer is then no longer empty, the read completes, and PA can continue executing. In the general case we have more than two members in the universe of processes. Some of the processes act as servers and others act as clients of the servers. (A process may be a server in some contexts and a client in others.) At any given time, therefore, several client processes may have outstanding requests to a given server process and may therefore be hanging awaiting their respective responses. The subsequent order of execution of these client processes will then depend upon the order in which responses arrive in the respective buffers.

An example of one server and many clients is the Dispatcher (server) and several concurrent diagnoses (clients). The diagnoses may concurrently send commands to (different) devices and therefore hang until the devices respond. When each device responds, the Dispatcher receives the response from TODOWN on the Sun and passes the result to the appropriate diagnosis process which then un-hangs and continues running.

#### 4.2.1.2 DEVICE SCHEDULING

Access to test/measurement devices must be scheduled in order to prevent conflicts and interference. A process in MITEC, the Scheduler, provides such device scheduling.

As processes execute in MITEC they may need communication access to devices. The multiplicity of processes and the asynchronousness of their execution may result in situations in which more than one process wants to communicate with a given device. Since some measurements require a sequence of commands, the device must be under the exclusive control of only one measuring process for the duration of the sequence. If another process were to break in with commands to the device, the sequence of measurement commands would be compromised. Another constraint is the need for control over several devices simultaneously in order to perform certain measurements. These requirements suggest the need for a mechanism whereby a process can request exclusive communication control over a set of devices. When given such control, the process holds on to the devices for as long as necessary and then releases control.

The MITEC Scheduler provides scheduling of exclusive communication rights for arbitrary sets of devices. Three calls are implemented:

1. Conditional request: This call requests that the set of device(s) be scheduled, unless one or more is already scheduled. An immediate return indicates whether the request was successful. If the device(s) were scheduled then the calling process can continue with its intended activities; otherwise, it presumably has some alternative actions planned.

2. Unconditional request: This call is the same as the conditional request except that the caller does not want to continue executing unless and until the device(s) are scheduled. To comply with this request the Scheduler checks the availability of the device(s): if available, they are scheduled and an immediate return to the caller is made. On the other hand, if one or more of the device(s) are already scheduled by another process, then the execution of the calling process is suspended until the device(s) are available.

3. Release device(s): This call releases the scheduled device(s), making them available to any other process that may want them.

The following example illustrates the Scheduler's role in obtaining waveforms from the digital oscilloscope. Three devices are needed for waveforms: the scope, the mini-matrix switch and the datalok. The need for the scope is obvious. The mini-matrix switch provides the circuit access and monitoring capabilities. Finally, the datalok is needed for selecting the pins where the signals are to be measured.

There are two classes of processes in MITEC that obtain waveforms, browsing processes and diagnosis processes. When a waveform is requested by an operator during browsing, the browsing process makes a conditional request for the three devices. If one or more of the devices is in use, the browsing process informs the operator that waveforms are currently unavailable; the operator is then free to browse elsewhere. On the other hand, a diagnosis process cannot be so flexible. It therefore makes an unconditional request for all three devices. If one or more of the devices is unavailable, it will hang until all three are available. It knows that its execution will continue when and only when the three devices are scheduled to it and therefore makes no allowance for alternative actions.

An area of device scheduling that has not been addressed in MITEC is the simultaneous scheduling of devices at two MITECs for a test or measurement that involves both sites. An example of this is a two-site BER test. The problem is that the Schedulers at the two MITECs operate independently of each other and therefore a deadlock situation can develop in which each site is waiting for the other to free up devices so that it can continue with its activities.

#### 4.2.1.3 SCHEDULER WINDOW

The MITEC Scheduler process maintains a window consisting of several panes which provide information relating to the current state of MITEC scheduling. Since this window is maintained dynamically by the Scheduler as MITEC executes, one can imagine watching this window as a way of monitoring MITEC's activities. The Scheduler panes are as follows:

1. A Resource Monitor pane that lists the devices that are currently scheduled. Next to each device are one or more blips (rectangles), each blip corresponding to a process that currently wants to schedule the device. The first blip corresponds to the process that actually has the device scheduled to it; the other blips, if any, indicate processes that have made unconditional requests for the device. When a device is completely released, i.e., is not scheduled to any process, it is removed from this pane.

2. A Message Log pane containing chronologically a line per each device request, device release, initiation of a diagnosis, and termination of a diagnosis. This pane is scrollable permitting one to review old actions.

3. A Station Log pane showing a blip for each diagnosis process currently in existence. These blips are mouse-sensitive; mousing any one provides basic information (in the Inspector Window, see below) about the corresponding diagnosis. The basic information contains such items as CCSD of the circuit, time the diagnosis started, and more.

4. An Inspector Window pane provides information that has been elicited by the operator from the other panes. Currently, the only information provided is basic information about a diagnosis (as described above in description of Station Log pane).

Figure 4.1 shows a copy of a scheduler window, as taken from the screen of a Symbolics terminal.

### 4.2.2 COMMUNICATION

#### 4.2.2.1 PROCESS TO PROCESS

At times, tasks or processes in MITEC need to communicate with other processes. This occurs especially when a process wants a service-process to perform an action for it. For example, a Diagnosis process wants to communicate with the Scheduler process to schedule device(s) before using them for a measurement. This section describes the handling of such communication in MITEC.

Since processes execute independently of each other, we must allow for the possibility of several processes wanting to communicate with a service-process at the "same" time. To handle such interprocess communication we use the keyboard buffer queuing mechanism provided by the Genera operating system on the Symbolics. In this mechanism, the service-process has an input queue from which it reads a message, performs the actions requested by the message, and then attempts to read another message. If there are no more messages in the queue, the process hangs until another message appears. When a client-process wants the service-process


Figure 4.1
Scheduler Monitor Window
for TCF: WEST

to perform an action, it places a message in the input queue of the service-process. Depending on the situation, the client then either continues with its normal processing or hangs waiting for a message in its input queue from the service-process indicating the completion of the requested action.

#### 4.2.2.2 PROCESS TO DEVICE

Communication between a MITEC process in the Symbolics and a device is handled by a MITEC Dispatcher, which acts as the intermediary between a process on the Symbolics and TODOWN on the Sun. A subroutine, command-equipment, is used by a process to tell the Dispatcher that it wants to communicate with a device. This subroutine takes several parameters, including the device-name and the command to be sent. The Dispatcher then arranges for the command to be sent to the device and for the response to be delivered to the caller.

To associate a commanding process with a device's response, the Dispatcher uses a message-id mechanism. It manufactures a unique message-id which it includes in the message sent to TODOWN on the Sun; TODOWN remembers this message-id and returns it with the response from the device. The Dispatcher then branches on this message-id to return the response to the calling process. A list of message-ids together with the commands involved is maintained by the Dispatcher to allow the operator to issue a manual request to repeat a command.

For development purposes, the Dispatcher maintains a window which consists of a scrollable log of the commands and responses, an area for viewing the detailed response to a command, and a menu. The menu enables one to select one of the following actions:

1. Obtain the status of TODOWN by sending a status command to TODOWN.

2. Manufacture a command to a device on the spot by supplying the device name and the command to be sent.

3. Open, close, or change parameters of the serial line between the Symbolics and the Sun.

4. Display commands that have been transmitted for which no response has arrived.

5. Modify the display of commands and responses.

#### 4.2.2.3 MITEC TO MITEC

Situations develop during diagnosis in which the diagnosis process wants the neighboring MITEC to perform a test or measurement and return the result. Later on, when a patch is necessary, the

diagnosis process needs cooperation with its neighbor to determine the suitability of a proposed patch and then to install the patch. These actions and others suggest the need for a MITEC-to-MITEC communication mechanism. In this section we describe this inter-MITEC communication.

The basic low-level communication devolves from the RS-232 connection between the Symbolics computer and the Sun computer. The additional requirement is a connection between a Sun computer serving one MITEC site and its counterpart at another site. This connection can be a simple RS-232 cable, as is the case at Lincoln where we have two MITECs in one room.

With the low-level connections in place, there are several stages involved in MITEC A (MA) sending a message to MITEC B (MB). MA sends a message to the TODOWN on its Sun computer for forwarding to MB. TODOWN parses the message, notices that it is destined for MB, looks up MB in its table, notices that the way to get to MB is by sending the message on its port that is connected to the Sun at the MB site, and sends the message out on this port. The message then arrives at the Sun at the MB site, is read and parsed by TODOWN there, and it then delivered to MB.

In principle, several processes on MA may wish to communicate concurrently with MB. The actions and requests directed to MB may legitimately be handled simultaneously by MB. Therefore, we felt no need to impose the command-response paradigm on inter-MITEC communication that we imposed on MITEC-to-device communication. From TODOWN's point of view, masters (MITECs) may send any number of messages to other masters without waiting for a response between messages. The Dispatcher uses an internal message-id system so that a message between two MITECs that is logically a response to a previous request can be associated with, and delivered to, the requesting process.

Messages that MITECs send to each other are basically requests for diagnosis actions. There is an internal protocol which specifies a test to be run, a query as to the availability of a channel for patching, a command to perform a patch, and others. Where appropriate, the requests in this category have matching responses which also use the internal protocol.

### 4.2.3 OPERATOR INTERFACE

#### 4.2.3.1 TERMINAL INTERACTIONS

The Genera operating system on the Symbolics provides a rich world for a program to interact with an operator. We exploited this environment in MITEC.

MOUSE

We use the mouse for selecting items from a menu. A variety of menu styles and formats are available. In the simplest kind of menu a list of items is presented to the operator who chooses the item of interest by positioning the mouse and clicking the appropriate mouse button. A more complex menu presents several multiple-choice lists to the person for choosing aspects of mode or program behavior characteristics. In this menu the "current" selections are shown in a different font (typically bold) to distinguish them from the other choices that are available. Variations of these menus and others are also used in MITEC.

We use the mouse for selecting items (textual or graphical) from a display. The following are several examples. In the detailed graphical display of a circuit the individual components are all mouse-sensitive: one can "mouse" a component in order to obtain more information or, in certain situations, perform tests and measurements relating to that component. The items shown in the history pane during a diagnosis are mouse-sensitive: mousing one of them provides expanded information (possibly including a waveform) about that stage in the diagnosis. Commands and responses in the Dispatcher may be moused in order to repeat a command or view a response in detail.

KEYBOARD

The heavy use of the mouse in MITEC has relegated the keyboard to minimal use. Probably the most prevalent use occurs after some information has been presented to the operator in a temporary window which will vanish when he is ready. The operator signals such readiness by hitting any key on the terminal.

4.2.3.2 PANES

The screen of the Symbolics terminal in MITEC is generally divided into several panes, or windows. Each pane has a size and a collection of display characteristics, e.g., whether it is scrollable. Constraint frames are collections of panes laid out in a specified arrangement. MITEC uses constraint frames to provide different configurations of panes for different stages in diagnosis and browsing. Since the contents of a pane stay with the pane wherever it is displayed, one can easily switch between

screen layouts without having to regenerate graphics or textual material.

4.2.3.3 GRAPHICS

In the early days of the previous project, ETC, we made a decision that graphics displays of circuits would be generated dynamically rather than from bit-mapped representations stored on the disk. Not only does this approach avoid use of huge amounts of disk space, but it also provides the flexibility of modifying the

appearance of displays by modifying the display-generation software without having to regenerate old stored displays. We have continued this philosophy unaltered in MITEC.

There are two major areas of displays which primarily involve graphics. One is waveforms obtained using the digital oscilloscope; this is described elsewhere in this document. The other area deals with displays of circuits and their components; this is described below.

The circuit graphics display software first calls upon a subroutine to obtain a list of the items that are to be displayed. Parameters to this subroutine include the CCSD of the circuit and whether the end-to-end list or the detailed local TCF list is desired. In the latter case, other parameters are the trunk expansion level desired and whether one or both "arms" of the circuit are to be displayed. Using the list of objects, the graphics software builds a list of display objects containing all the information needed for display. By evaluating the sizes of the objects the software is able to determine the extent to which the resulting display would be crowded. One of three sizes of icons and text is then chosen to minimize crowding and maximize readability. (Size selection can be manually overridden by the operator.)

#### 4.2.3.4 MODES AND OPTIONS

In many situations in MITEC an operator may specify modes and options relating to program execution. Some of this flexibility devolves from the needs of the developers of MITEC. Others result from the desire to provide a rich set of execution choices. The following is an example.

One may specify whether diagnosis is to proceed in lock-step mode or automatically. In lock-step mode the diagnosis halts at strategic stages waiting for the operator to mouse "continue" before proceeding. This mode allows for demos at a leisurely and understandable pace; it also allows the operator to mouse "abort" if he does not agree with the line of reasoning being used by MITEC to diagnose the fault.

Typically, after several diagnosis scenarios have been demonstrated in lock-step mode we usually turn lock-step off and allow diagnosis to proceed at full speed in order to demonstrate MITEC's speed.

### 4.2.4 OPERATOR INTERFACE APPLICATIONS

This section describes applications of the operator interface tools in the areas of browsing and diagnosis.

#### 4.2.4.1 BROWSING

Browsing the database is a major subsystem of MITEC. The following

describes some of the characteristics of this subsystem.
When one selects Browsing one obtains a menu which lists the various types of items available in the database. One can then mouse one of the items and obtain a new menu listing the specific items of that type that are included in the database. At this point one mouses the specific item that one wants to see and obtains a display showing the item graphically, if possible; some textual information about the item; and a menu offering additional information and actions. At any point one may return to an earlier menu and make another choice.

The kind of information provided during Browsing depends upon the item being browsed. A standard set of icons for various kinds of objects handled by MITEC is available and is used, where practical, to show graphically the item being browsed. Textual information is provided for such items as fpi-location, data-rates, signal-level, etc. Tabular information relating to the characteristics of the ports of a multiplexer are available by mousing "show device info" in a menu. Finally, one can mouse menu items to see the LISP language representation of the object in the database (of use only to the developers) and to exit the browsing of this object and return to the previous menu.

Browsing a circuit provides an end-to-end display of the circuit in one pane and a detailed view of the circuit from the point of view of the local TCF in another pane. The size of the graphics is chosen by the software (as described earlier) and the detailed view shows full trunk expansion and all "arms" of the circuit. These default choices may be overridden by the operator via a menu which provides other options. Additionally, the objects in the detailed view of the circuit are mouse-sensitive; mousing them provides additional information, measurements, or waveforms, as appropriate.

Browsing a Datalok does not produce graphics. Instead, a set of commands is sent to the device to obtain its current relay settings which are then presented. The combination of relay settings is interpreted into the pin number(s) involved.

Browsing a mini-matrix switch does not produce graphics but instead provides the current DCE-to-DTE map from the switch's memory.

Figure 4.2 shows a sample copy of the appearance of a Symbolics screen when one is browsing a circuit.

#### 4.2.4.2 DIAGNOSIS

Diagnosis makes heavy use of the operator interface facilities in order to interact with the operator and to show the progress of the diagnosis.

One enters diagnosis by mousing "diagnose fault" in the top- level MITEC menu. A new menu appears giving the operator one of the following choices.

1. Return to the latest diagnosis to continue where it left off.

2. Enter a complaint and thereby start a new diagnosis.

If one mouses "enter user complaint", one obtains a menu of all the possible complainers, i.e., the end-users and matching circuits that MITEC knows about. The operator mouses one of the choices and thereby enters into a diagnosis. (Currently, the complaint is assumed to be "no signal" and therefore there is no provision to specify a type of complaint.)

As one enters diagnosis one is presented with a scope configuration consisting of a menu of utility and development functions (not described further) and five panes. The panes are:

1. Basic static information about the fault being diagnosed.

2. Messages describing the diagnosis as it is running including tests that are about to be run, results of tests, conclusions, and other narrative information. This pane is scrollable, permitting one to review past information. If lock step is on, this pane instructs the operator when to mouse "continue".

3. History of the diagnosis in the form of a line for each measurement that was performed and each communication that was done with a remote MITEC. Each of the history lines is mouse-sensitive; mousing any of them produces an expansion of the information associated with that line. Such expansion includes a re-display, if appropriate, of the corresponding measurements and waveforms that were previously obtained.

4. A detailed display of the circuit being diagnosed. The items in this display are mouse-sensitive; mousing any one produces additional information about it. This display also contains a dashed outline around the item that is currently the "focus of attention" in the diagnosis. (This pane also appears when one is browsing a circuit.)

5. An end-to-end display of the circuit being diagnosed. (This pane also appears when one is browsing a circuit.)

Management of the panes during diagnosis is implemented by three interacting processes: history, graphics, and fault isolation. As such, actions involving these processes may proceed in an asynchronous fashion.

The history process manages the contents of the history pane. It handles the mousing of a line in the pane by producing an expansion

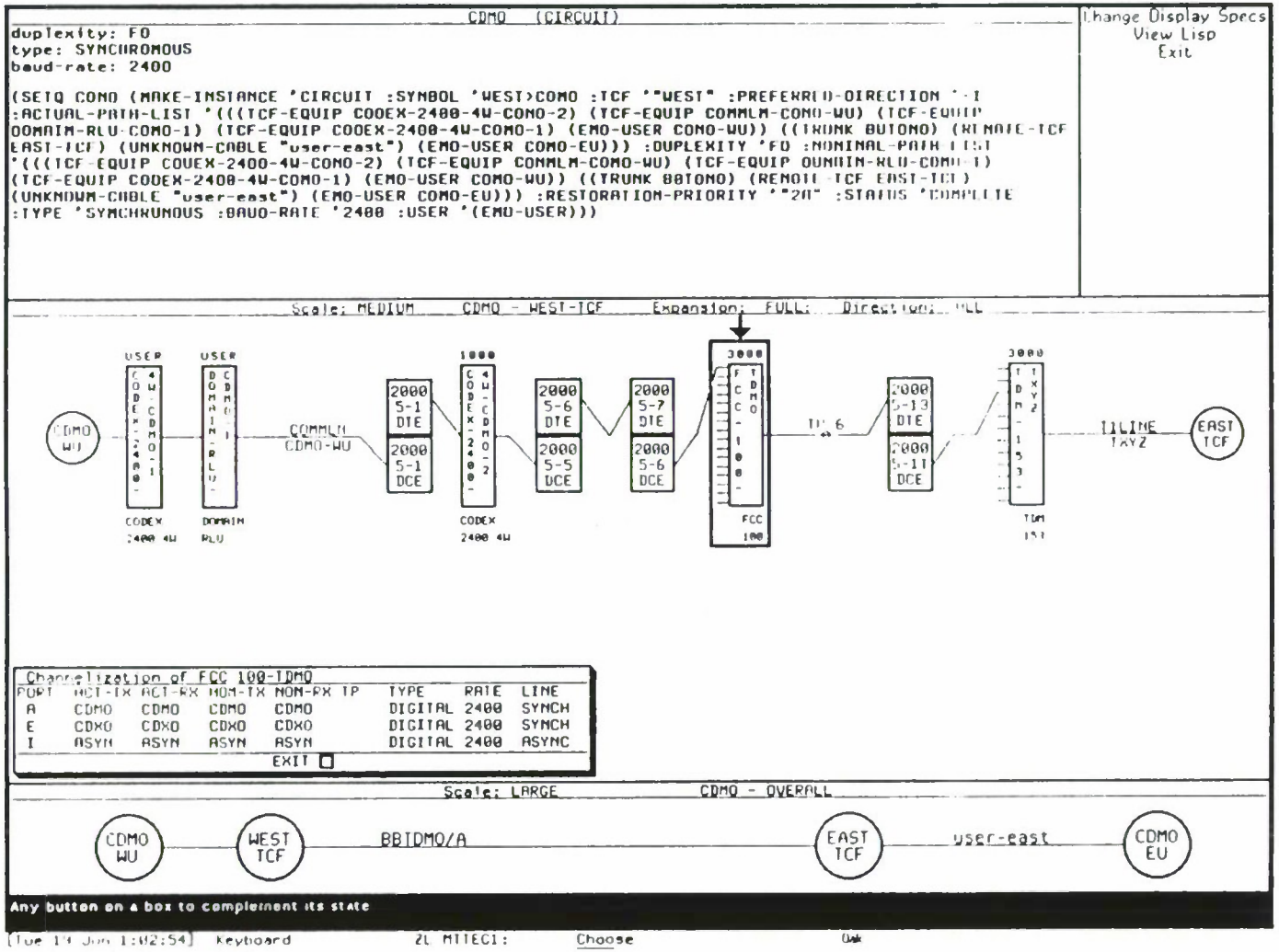



Figure 4.2
CDMO (Circuit)

of the information relating to that line; in some cases the expansion involves calling up an old waveform for redisplay.

The graphics display process generates the detailed circuit display and the end-to-end circuit display. It also handles the mousing of objects in the detailed circuit display, calling the appropriate subroutines to supply the additional information about the moused item. It also displays the "focus of attention" dashed outline around the appropriate item.

The fault isolation process is the master diagnosis process and handles the other panes as well as the fault isolation strategy. During initialization it generates the static-information pane. Later, it maintains the message pane and the flow of output to it about tests, measurements, etc. History is maintained for display in the history pane.

Figure 4.3 shows a sample copy of the appearance of a Symbolics screen when one is diagnosing a circuit.

### 4.2.5 Database

#### 4.2.5.1 Purpose

A MITEC database contains specific knowledge about the circuits and equipment belonging to a TCF. This models the distributed knowledge among Tech Controls today. When DCA issues a telecommunications order, each TCF involved knows the overall circuit connectivity. The knowledge about the detailed circuit path is distributed among the TCFs; each one knows only the details pertaining to its facility.

The database information is distinguishable from the MITEC software which performs the various TCF operations. Software separation of the TCF-specific circuits and devices from the general TCF functions means that MITEC is not implemented for any particular TCF. In order for MITEC to be used in a TCF, there must be a database, MITEC must be able to represent the different types of equipment and circuits in the TCF and know the correct procedures to be used.

#### 4.2.5.2 Organization

The MITEC database can be thought of at two different levels. Since we are not using a commercial database tool, there is a lot of higher-level software which establishes a framework for organizing and maintaining the data. This higher-level software is not tailored to a specific TCF but can be thought of as MITEC's database management system. Most of the discussion which follows refers to the higher-level database software, although some examples of TCF-specific data are given.

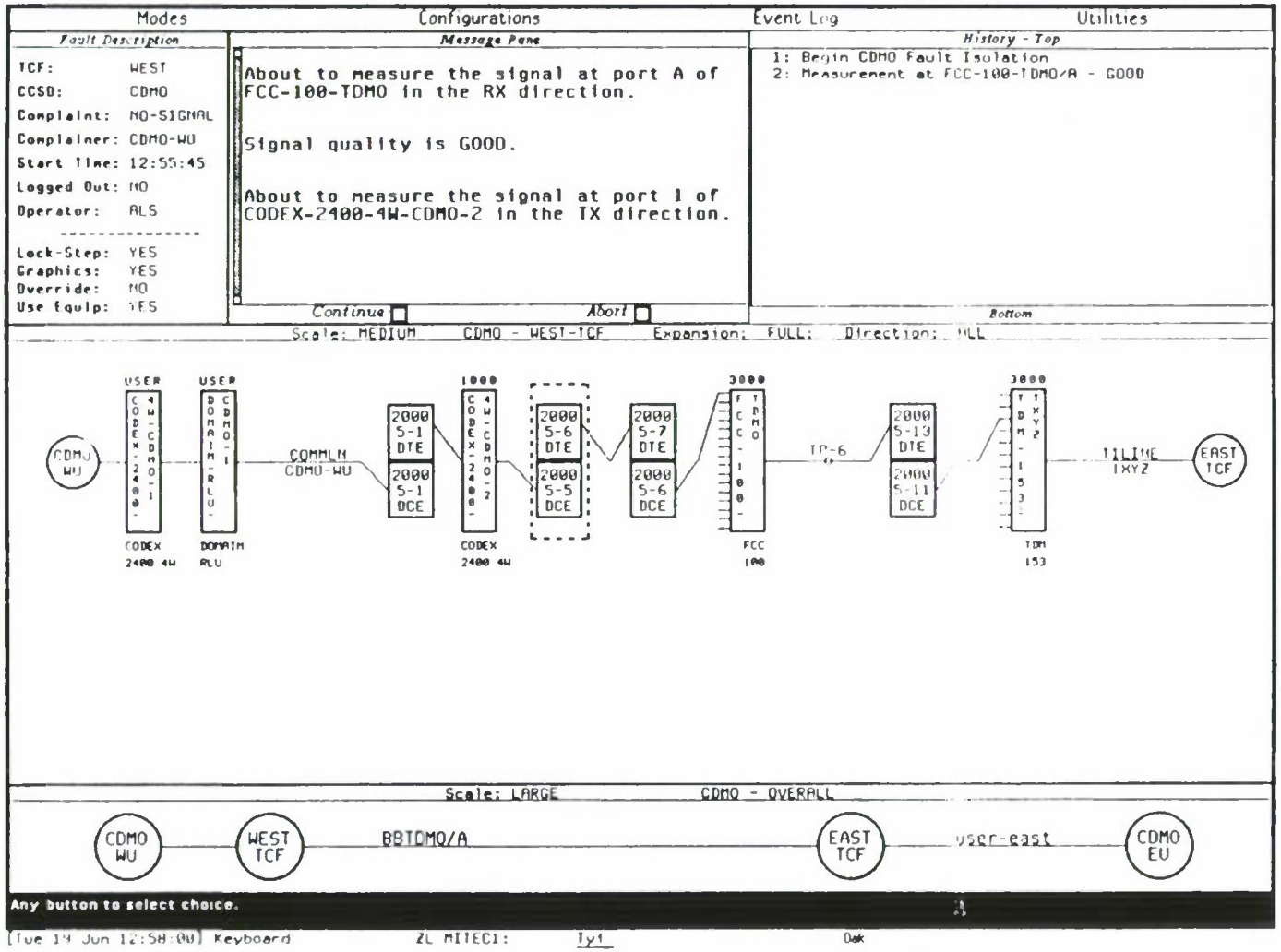



Figure 4.3
MITEC Display During a Diagnosis

MITEC's database is object-oriented and hierarchically organized. By object-oriented we mean that we can define an object-type which has a set of known attributes. Default attribute values may or may not be specified. We can also define a set of operations that can be performed on a class of objects, i.e., all the objects of a certain object-type. Object classes can be built upon other defined object classes resulting in an hierarchical knowledge structure. In this way, objects inherit properties from a parent object. An object may have several parents and, thus, inherit properties from each parent.

Devices and circuits are the two major data categories in the MITEC database. These categories inherit features from predefined object-types but they also provide the basis for defining new object-types. Figure 4.4 shows the hierarchy built upon the device class and Figure 4.5 shows the circuit hierarchy. The TCF-specific knowledge is not shown but consists of objects defined by the types in the leaf nodes.

#### 4.2.5.3 Implementation

The MITEC database implementation makes extensive use of Symbolics Zetalisp flavors as both a means of defining object types and hierarchically structuring the object types. Each equipment and circuit item in the MITEC database is a lisp-form of the following pattern:

```
(setq <object-name> (make-instance <object-type> :property-1 value-1...))
```

There is a unique object-name for each item in the database. Objects are named only because they ease development; names are not necessary. The object-type is a predefined object class to which the object belongs. The properties are inherent to the object-type whereas the values may be different for each object in that class of objects.

Examples from a database for the MITEC West testbed are shown in Figures 4.6 - 4.9. Circuit CDMO (Figure 4.6) and trunk BBTDMO (Figure 4.7) both have the same parent class of circuit giving them the same property of "nominal-path-list". However, the examples show that they inherit drastically different properties from their other parent objects. The same type of similarities and differences can be observed with multiplexer FCC-100-TDMO (Figure 4.8) and modem CODEX-2400-4W-CDMO-1 (Figure 4.9), both instances of the device parent class.

The database is implemented as a collection of I-O (instance-of) files which reside on the Symbolics disk. Each I-O file contains objects which are in a given object class. For example, in the file "i-o-fcc-100", there are descriptions of each FCC-100 which appears in a particular TCF. These I-O files are used to generate a second collection of files called DB files. The purpose of

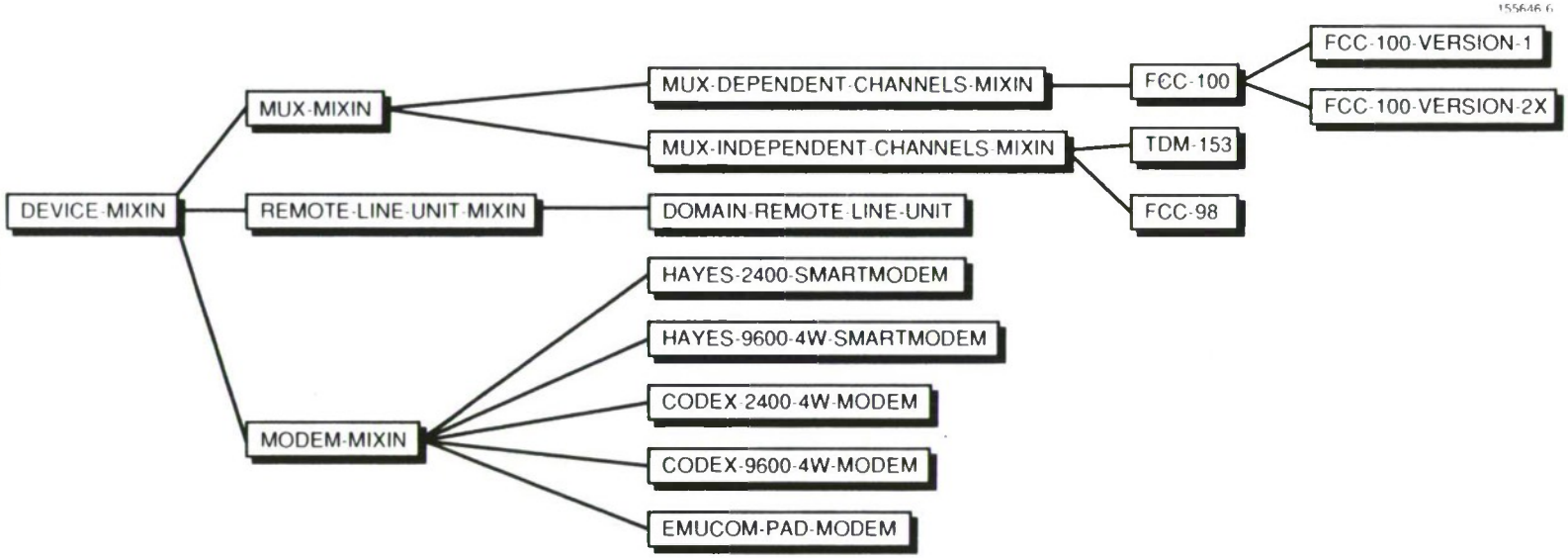



Figure 4.4
Device Feature Hierarchy

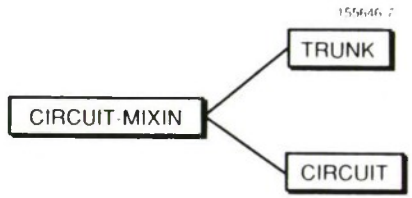



Figure 4.5
Circuit Feature Hierarchy

```
Jun 19 09:43 1990  db-examples Page 1

-*- Mode: Text -*-
Figure 4.2.2.3    MITEC West Circuit CDMO

(setq cdmo
      (make-instance 'circuit
                     :symbol 'cdmo
                     :duplexity 'fd
                     :preferred-direction -1
                     :nominal-path-list '(
                                          ((tcf-equip codex-2400-4w-cdmo-2)
                                           (tcf-equip commln-cdmo-wu)
                                           (tcf-equip domain-rlu-cdmo-1)
                                           (tcf-equip codex-2400-4w-cdmo-1)
                                           (end-user cdmo-wu)
                                           )
                                          (
                                           (trunk bbtdmo)
                                           (remote-tcf east-tef)
                                           (unknown-eable "user-east")
                                           (end-user cdmo-eu)
                                           )
                                          )
                     :type 'synchronous
                     :baud-rate 2400
                     :status 'complete
                     :user '(end-user)
                     :restoration-priority "2a"
                     )
      )
```

Figure 4.6
Circuit CDMO Database

Figure 4.2.2.4  MITEC West Trunk BBTDMO

```
(setq bbtdmo
      (make-instance 'trunk
                     :symbol 'bbtdmo
                     :duplexity 'fd
                     :preferred-direction -1
                     :aggregate-circuit
                     '((actual-tx tdmo) (actual-rx tdmo)
                       (nominal-tx tdmo) (nominal-rx tdmo))
                     :carry-circuits
                     '(
                       (a (actual-tx cdmo) (actual-rx cdmo)
                        (nominal-tx cdmo) (nominal-rx cdmo))
                       (e (actual-tx cdxo) (actual-rx cdxo)
                        (nominal-tx cdxo) (nominal-rx cdxo))
                       (i (actual-tx asyn) (actual-rx asyn)
                        (nominal-tx asyn) (nominal-rx asyn))
                       )
                     :nominal-path-list
                     '(
                       ((tcf-equip fcc-100-tdmo)
                        (circuit tdmo)
                        (remote-tcf east-tcf)
                        )
                       )
                     :status 'complete
                     :type 'digital-trunk-circuit
                     ))
```

Figure 4.7
Trunk BBTDMO Database

Figure 4.2.2.5  MITEC West Device FCC-100-TDMO

```
(setq fcc-100-tdmo
      (make-instance 'fcc-100-version-2x
                     :symbol 'fcc-100-tdmo
                     :comm-name "wfcc"
                     :ports '(
                              (far 6 (telenex-switch-1 8)
                                   digital-mil-188-balanced 56k synchronous)
                              (a nil (telenex-switch-1 5) digital-rs-232-unbalanced 2400 synchronous)
                              (c nil (telenex-switch-1 14) digital-rs-232-unbalanced 2400 synchronous)
                              (i nil (telenex-switch-1 28) digital-rs-232-unbalanced 2400 asynchronous)
                              )
                     :connected? t
                     :fpi-location "3000"
                     :device-label "fcc-100"
                     :committed-composite-rate 7200
                     :sys-channel-rate '(nrz 56k buf4 pos-mark ext-clock frame1 rcm-hw)
                     :channel-rate '((syn 2400 neg-mark ext-clock) (0) (0) (0)
                                     (syn 2400 neg-mark ext-clock) (0) (0) (0)
                                     (asy 2400 ncg-mark nil 8bit 1stop)
                                     (0) (0) (0) (0) (0) (0) (0))
                     ))
```

Figure 4.8
FCC-100-TDMO Database

Figure 4.2.Z.6  MITEC West Device CODEX-2400-4W-CDMO-1

```
(setq codex-2400-4w-cdmo-1
      (make-instance 'codex-2400-4w-modem
                     :symbol 'codex-2400-4w-cdmo-1
                     :ports '(
                              (far nil () vf 2400 synchronous)
                              (1 nil (domain-rlu-cdmo-1 far) digital-mil-188-unbalanced 2400 synchronous)
                              )
                     :fpi-location "user"
                     :device-label "codex-2400 4w"
                     :nominal-send-signal-level '(0 3)
                     :nominal-receive-signal-level '(-16 3)
                     :remotep t
                    ))
```



Figure 4.9
CODEX-2400-4W-CDMO-1 Database

having two sets of database files is to preclude the possibility of polluting or totally destroying the database during software development. The I-O files contain permanent knowledge and are not modified by the MITEC software. Hence, the I-O files represent a pristine hardware configuration. The DB files are what could be updated to reflect any circuit patches or changes in device states that the MITEC software controls. Currently, modifications exist only in the memory of the Symbolics and are not written onto the disk.

Rather than cluttering the Symbolics memory with the entire TCF database, MITEC reads only those database objects into memory that it needs. When the DB files are generated from the I-O files, equipment and circuit objects are read from the I-O files and a hashing scheme is used to associate the object's name with a DB filename where the object will reside. This mechanism evenly distributes the objects over a collection of files and provides an approximately uniform search time through a file for a particular object. This scheme has worked well during the MITEC development and could be used with both small and large databases.
While building the DB files, MITEC dynamically generates three lists: a db-item-list, a resource-list and a circuit-destination-list. The db-item-list is a list of all the different devices and circuits in the database organized by object class. For example, a category called "electronic-patch-box" is associated with the names of all the electronic patch boxes in the TCF. The resource-list provides an association between the communication equipment and the circuits riding the equipment. The circuit-destination-list, which groups a destination with all the circuits going to that destination, makes it easy to locate patching and preemption options for a failed circuit. Since the lists are generated dynamically, updating multiple lists is not an issue when adding new objects to the database.

#### 4.2.5.4 ISSUES

The MITEC database implementation is sufficient for prototyping the MITEC concept. The hierarchical, object-oriented techniques are extremely appropriate for representing and manipulating the TCF-specific data. However, if viewing MITEC as a deployable system, there are several issues which have not been addressed. Most of these issues are resolved by commercial database systems which are developed for multiple users. At the end of this section, we describe two issues which are not solved by commercial database systems.

Modifications to an object are currently not written onto disk. Once an object is accessed from the database the object remains in Symbolics memory. Modifying an object changes the object representation in memory. This means that if the machine crashes, all the information reflecting an object's changed state is lost. In an operational MITEC, database changes would indeed be written

out to disk. Furthermore, if a change requires updating two or more objects, there must be some mechanisms which, in the event all objects cannot be updated on disk, prevents only part of the transaction being completed.

The MITEC database can be accessed by concurrent processes. There is no mechanism being used to establish read-write privileges on objects. Nothing prevents one process which starts to change an object from being swapped out by the scheduler before the modification is complete. Two processes should never be trying to write modifications to the same object at the same time.

There is no concept of backing up the database in MITEC that would carry over to an operational system. A deployable MITEC needs to have hardware facilities and software utilities designed for this purpose.

Our research effort has not studied scaling up the size of the database to meet the needs of an entire TCF. The following are examples of unanswered questions: What size disk is needed? How fast must transactions occur? How many transactions are needed for each operation?

A user-friendly database entry mechanism is the first issue not resolved by a commercial database. In MITEC, objects are entered into the database by editing files. The fielded MITEC not only would need a robust user interface for data entry but also should verify the contents of each field when possible. There should also be consistency-checking throughout the whole database. For example, an operator should not be allowed to define two circuits which ride the same port of a device.

The second issue which is not solved by resorting to a commercial database system involves maintaining the consistency between the MITEC database and those devices with their own memory. There is always the possibility that a device goes down due to a power outage and comes back up in a state different from the state MITEC associates with the device, or vice versa. If a device is controllable from a front panel then operators must not change the device state without telling MITEC. Preventing and dealing with MITEC-device inconsistencies is a major challenge for MITEC.

### 4.3 MITEC FAULT ISOLATION AND RESTORAL

Troubleshooting and restoration procedures have been designed and implemented in MITEC for point-to-point single user and trunk circuits. The ability to electronically access the circuit and communicate directly with test equipment enables MITEC to analyze signal behavior and perform restoral actions independent of an operator. In July 1989 at a MITEC Steering Committee meeting, we demonstrated most of the fault isolation capabilities developed during FY89. The results of these demonstrations indicate that it

is possible to automate a majority of tech control fault isolation and restoral procedures. Most significant is the reduced time required by MITEC, in contrast to the human, to perform independent troubleshooting tasks. Clearly, as more fault isolation procedures are automated, troubleshooting will become more accurate and Tech Controllers will be freed to work on more complex tasks.

The signal tracing strategy applied in MITEC largely developed from our interactions with Andrews 2045th Telecommunications Group during a previous project, the Expert Tech Controller (ETC). The strategy is compatible with current Tech Control practices and adheres closely to the goal of restoring communication services as soon as possible in the event of failure. Our implementation to-date has focused on loss-of-continuity problems with digital, synchronous circuits. To the extent possible, we have generalized the fault isolation knowledge in MITEC so that the same methodology can be applied to all circuits. When tests or strategies should vary depending on the circuit type or equipment, we depend on the database for supplying the discriminatory information.

Over the span of the MITEC project, a great deal of knowledge has been gained regarding fault isolation and circuit restoral that is yet to be implemented. Some of those observations are documented here as a means of retaining useful insights and providing guidance for future MITEC development. Notes specifying MITEC's current implementation status are frequently provided.

### 4.3.1 Fault Isolation Phases

#### 4.3.1.1 Trouble Detection

The critical first step in fault isolation is trouble detection. MITEC fault isolation is currently initiated by external sources: a complaint from an end-user, a request from a distant-end TCF, or an equipment alarm. Eventually, the collection of data from standard quality control testing will enable MITEC to initiate fault isolation based on its own initiative. There are several issues regarding trouble detection which a deployed MITEC system must deal with effectively. MITEC's current implementation deals with each area to a varying degree.

Depending on the direction(s) of the problem, one or both end-users may be affected. If both end-users notice a problem and complain to their local TCFs then both TCFs will start working on the same problem. It is foreseeable that two MITECs could get into a situation where they are both waiting for the other to complete a task on the same circuit. Consequently, both MITECs may reach a deadlock state while in contention for test equipment. MITEC currently does not attempt to coordinate problems with another MITEC.

The occurrence of an equipment alarm can result in multiple alarms from the same piece of equipment or different pieces of equipment which are interconnected. How alarms affect one another depends on the type of alarm, the type of equipment and how the equipment involved is currently configured. For example, if there is an alarm indicating a failed clock from one device then another device that is deriving its clock from the failed device may alarm or it may have a mode which starts generating an internal clock and continues to work fine.

In the current MITEC design, "interesting" equipment alarms are connected to a Datalok relay panel. The Datalok is capable of detecting a change of state on any of its relays. This change of state information is collected on all the relays over a given time interval. When MITEC requests this information from the Datalok, MITEC is able to determine the current state of an alarm and whether an alarm has been frequently changing state.

MITEC must know how to interpret and respond to the alarm information from the Datalok. Alarms indicate different types of problems: timing problems between devices, power failure, and change in operational mode. While some alarms indicate serious problems that require troubleshooting, alarms do not consistently indicate failure. Some alarms can be anticipated based on current MITEC actions. Equipment alarms can also precede user complaints. MITEC needs to be able to relate one complaint to another so that all problems are indeed resolved and in an organized way.

Complicating the issue of multiple alarms even more, is the fact that alarms don't always occur simultaneously. When MITEC receives an equipment alarm, it must first check to see if the alarm relates to any other troubleshooting scenarios already underway. If the alarm does relate, the relationship between the alarms must be understood so that MITEC can prioritize its troubleshooting procedures. If the latter alarm is thought to be the cause rather than an effect, then the process spawned to diagnose the first alarm will be suspended and another process spawned to diagnose the latter alarm. After the higher-level alarm is resolved, the lower-level alarm process should be resumed and a check made that the lower-level alarm is also resolved. If the lower-level alarm did not go away, then perhaps another problem exists and needs to be diagnosed.

#### 4.3.1.2 Troubleshooting and Restoral Strategy

After detecting a fault, TCs have the responsibility to restore service as soon as possible. This means that the first goal of fault isolation is to identify the faulty segment of a circuit path which can be patched around. Once service has been restored and time is not the critical issue, TCs can examine the faulty segment and attempt to pinpoint the source of trouble. MITEC incorporates these goals into its fault isolation design.

There are several different signal tracing strategies described in DCA Circular 310-70-1 and employed by Tech Controllers. The strategy MITEC currently uses was carried over from ETC; originally developed as a result of our interactions with Andrew's 2045th Telecommunications Group. First, the problem space (a circuit path) is divided into upstream and downstream circuit segments. Since the downstream TC is in charge of troubleshooting a fault, the MITEC strategy first tries to establish whether the upstream circuit segment is functioning properly. Once MITEC identifies the circuit segment with the problem, the first phase of troubleshooting continues until a point where circuit restoral is possible.

MITEC traces the signal from the point where trouble is detected upstream toward the source of the signal which is failing to be received correctly. The signal characteristics are observed at electronically accessible points using appropriate tests and equipment for the signal and complaint type. Analysis of the observed signal is used to determine which part of the segment is suspect and what conclusions can be drawn. Since coordination of user complaints and equipment alarms is so important, MITEC needs to identify whether a problem involves a single user only or involves multiple users. As the signal tracing progresses, MITEC may interact with the upstream TCF/MITEC which is in the circuit's path. For any given task, MITEC-to-MITEC interaction follows the master-slave relationship established in TCF procedures with the downstream TCF as master. When a faulty segment is identified, the next step is to try to restore circuit service. TCFs frequently have spare channels on a mux or entire spare systems dedicated for circuit restoral plans. Whether or not circuit service is restored in this first stage depends on whether a dedicated spare route is available.

The next stage of fault isolation attempts to pinpoint the fault in the circuit's path. Tests are generally more complex in this stage, frequently substituting channel cards or looping back a test tone at some point and looking for bit errors. Cooperation from the human operator and the distant-end MITEC are typically necessary. If the circuit service was not already restored, either the users are notified that the circuit will be down indefinitely or lower priority users are preempted. When the faulty equipment is repaired by maintenance personnel, circuits are restored to their nominal path.

### 4.3.2 Troubleshooting Circuits Between MITECs

One interesting aspect of troubleshooting is the cooperation required between TCFs. When a failed circuit comes to the attention of a TC, the immediate goal is to establish if the problem source is in-house or at another TCF. If the problem is in-house, then it is the responsibility of the local TCF to fix the problem. On the other hand, if the problem appears to be in

another TCF, then the local TCF must prompt the upstream TCF to look at the circuit signal. The upstream TCF now goes through the same procedures as the downstream TCF, possibly handing the problem off to the next upstream TCF.

Two general rules are observed in troubleshooting between TCFs. First, the downstream TCF is always in charge of troubleshooting.

Second, no neighboring TCF is ever skipped over in the point-to-point troubleshooting process.

One scenario which we have implemented in MITEC illustrates such MITEC-to-MITEC cooperation. See Figure 4.10 for a layout of the circuit described. A fault, simulating a failed FCC-100 channel card in East, is introduced into a circuit by opening the receive data signal between the FCC-100 port and the Telenex matrix switch. This causes the West end-user of the circuit to complain that they are not receiving a signal. MITEC West follows the signal tracing strategy and tests the signal at the most upstream access point where the circuit can be seen in its nominal form. MITEC West observes via the Telenex monitor port and digital oscilloscope that the signal is not being received correctly. (How MITEC measures signals is described in later sections.) MITEC West then asks MITEC East, the upstream TCF in this circuit, if East is transmitting a signal. MITEC East starts its own fault isolation process and determines that a good signal is being transmitted to West on the specified circuit. Upon receipt of verification from East, West (the downstream TCF in control) resumes troubleshooting. Since the circuit rides a multiplexer, MITEC West must determine whether or not a higher-level trunk has failed. A channel-level fault is indicated because there are no multiplexer alarms and no other circuits on the trunk seem to be in trouble. TCFs want to restore service as soon as possible to the circuit end-users. MITEC West searches for a spare channel on the same trunk and verifies with East that a channel card is available. Once West and East agree on the channel to be used, both MITECs make the necessary patch and service is restored.

### 4.3.3 Troubleshooting Tail-Circuits

Another circuit segment requiring special troubleshooting attention is the tail-circuit (also called an on-base circuit), i.e., the section of the circuit from the TCF to the end-user. The local TCF is responsible for both the end-user's communication services and equipment. Fault isolation procedures differ somewhat because the end-user is not equipped with test equipment or the know-how to perform tests like a remote TCF. A communication line, the user's modem and the user's terminal are typical out-of-house equipment in a digital tail-circuit. The communication line may or may not be leased and, perhaps, separate maintenance contracts exist for each of the modem and terminal. In these situations, there is not usually any patching capability. The troubleshooting goal is to

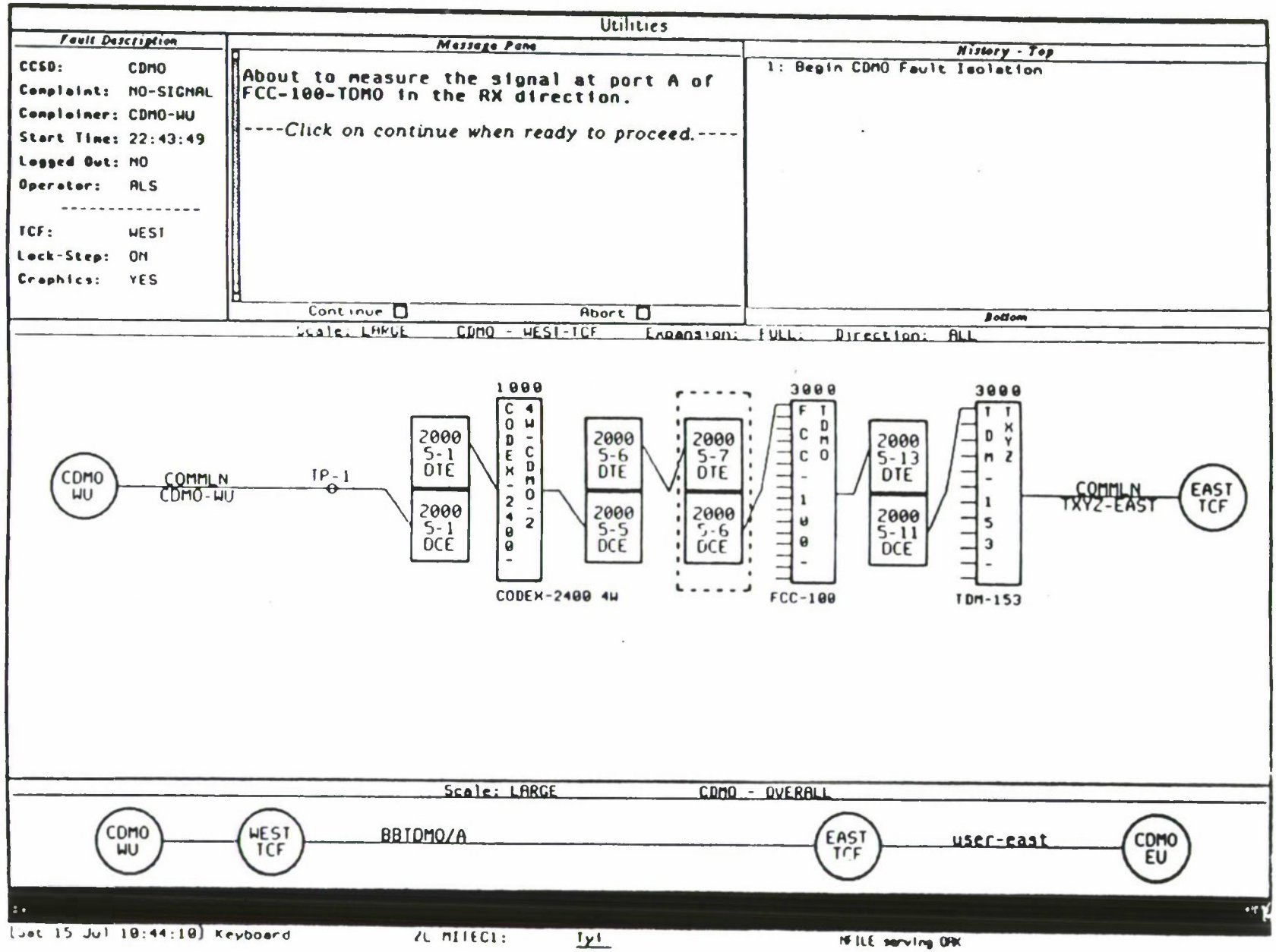



Figure 4.10
CDMO Circuit Layout Display

exactly identify the failed component so that the correct maintenance team can be dispatched.

An example of MITEC troubleshooting tail-circuits was demonstrated in July. See Figure 4.11 for a layout of the circuit described. We introduced a fault by disrupting the receive signal on the user's data source, the pocket BERT. The user's complaint of no-receive signal is received by the MITEC operator and provided as input to MITEC. The signal tracing strategy begins, as described previously, with MITEC observing whether or not a good signal is being received from the upstream TCF. In this scenario, an adequate digital signal is observed and MITEC deduces that the problem is in the tail-circuit segment. MITEC steps through the circuit's access points and observes the quality of the transmitted signal. After clearing its house, MITEC tests the communication line between the TCF and the end-user's site. A remote-line unit (RLU) on the user's premises allows MITEC to send a DTMF tone which commands the RLU to loop back the signal received over the communication line. MITEC then injects a test tone of known frequency and compares the actual level of the looped-back signal with the expected value maintained in the database. The communication line has an acceptable level, so MITEC concludes that the problem is in the user's equipment. MITEC has no way of automatically testing whether the user's modem or user's data source is at fault.

### 4.3.4 SIGNAL QUALITY ASSESSMENT: OSCILLOSCOPE FOR DIGITAL SIGNALS

#### 4.3.4.1 INTRODUCTION

We assume that the diagnosis process has determined that it needs to assess the signal quality at some point in a digital circuit and that the oscilloscope is therefore the appropriate device to perform the measurement. The diagnosis process then calls the scope module in MITEC with the appropriate parameters and requests it to obtain the specified waveform(s), analyze them, and return to the caller an assessment of the signal quality. Based upon whether the signal quality is good or bad, the diagnosis process will proceed accordingly.

This section provides a description of the mechanics of obtaining and analyzing the waveforms(s). Other sections elsewhere in this document describe the actual mechanics of communicating with the scope.

#### 4.3.4.2 WAVEFORM PARAMETERS

We now describe the parameters involved in selecting the measurement and analysis that are to be done. (This list describes the controlling factors and is not to be construed as the parameters in a calling sequence to a subroutine.)

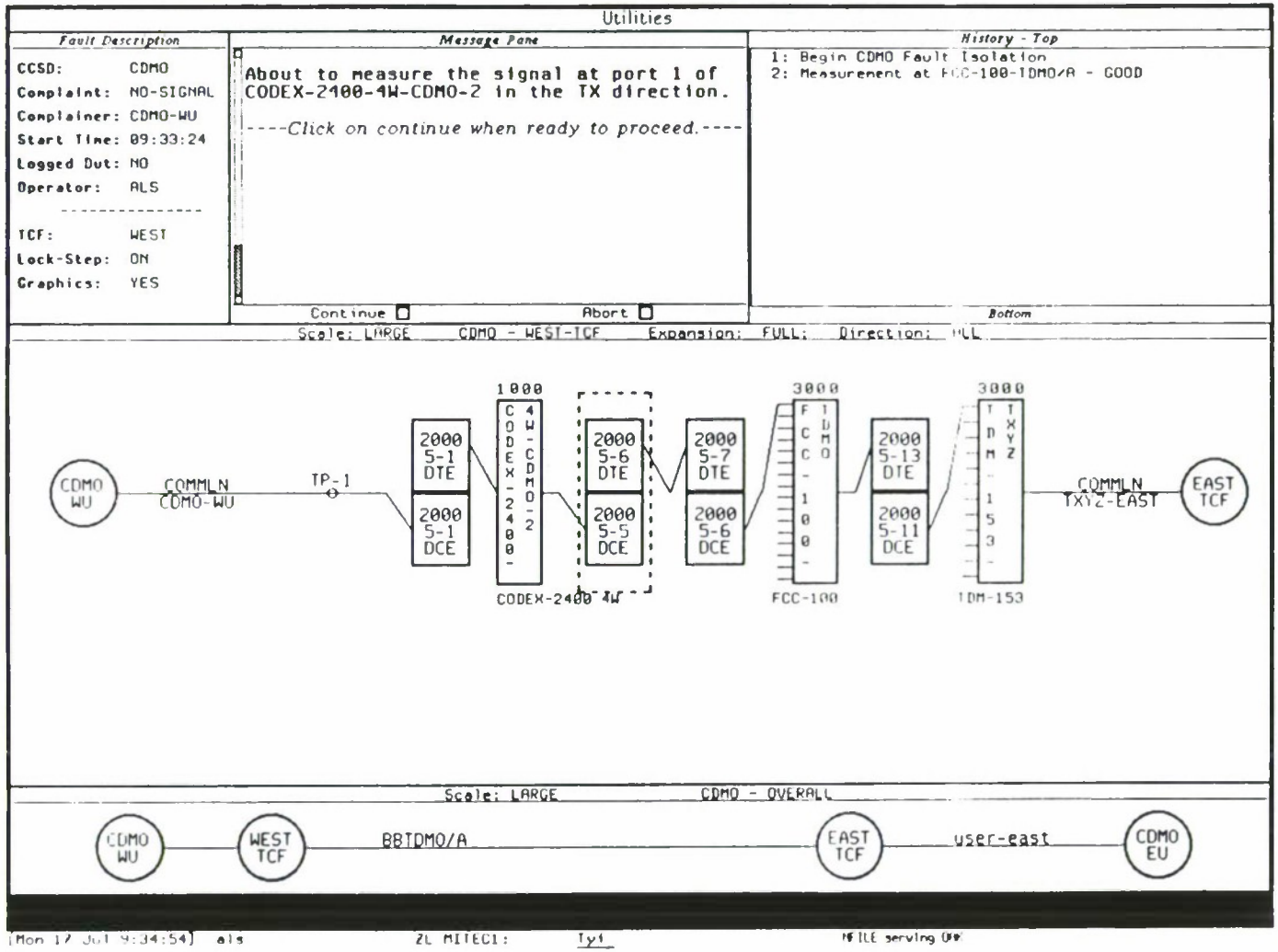



Figure 4.11
Tail Circuit Display

1. The location in the circuit where the measurement is to take place.

2. The complaint which caused the diagnosis to start and thereby motivates the measurement. Currently, "no-signal" is the only complaint that starts a diagnosis. Other possible complaints that may be handled in the future include "garble" (transmission is garbled), "receive-mark" (receiving only mark), "receive-space" (receiving only space), and others. This parameter is used by the scope module to choose the appropriate quality tests to be run on the waveform(s).

3. The type of signal that is being measured. Currently, "digital-rs-232-unbalanced" is the only one that is handled by a diagnosis. This parameter is used by the scope module to choose in a table-driven fashion the polarities, thresholds, pin numbers, etc. for the tests that are run on the waveform(s). This parameter is also used to derive a meaningful minimum and maximum expected amplitude value and to then specify to the scope an appropriate sensitivity value to use for the subsequent waveform(s).

4. The baud-rate of the signal. The scope module uses this parameter to choose an appropriate sweep-rate for the waveform(s) to be obtained and to test whether the mark to space (and vice versa) transitions in the clock and data signals are appropriate for the baud-rate. Sweep-rate relates to the amount of time encompassed by a given amount of horizontal displacement in the waveform. A faster sweep-rate provides finer detail while a slower rate compresses the waveform.

5. The measurement to be performed. Examples are "rx-data-clock" and "tx-data-clock" which involve the "data-plus" and "clock-plus" signals in the receive and transmit directions, respectively. This parameter provides information that is used in getting the waveform(s) and in displaying them. In particular, this parameter indicates the specific signals that are to be measured. Currently, there are ten signals that may be measured individually or in pairs. These signals and the pins on which they appear on an RS-232 line are as follows:

```
pin  2: tx-data-plus
pin 14: tx-data-minus
pin 24: tx-clock-plus
pin 23: tx-clock-minus
pin  3: rx-data-plus
pin 19: rx-data-minus
pin 17: rx-clock-plus
pin 18: rx-clock-minus
pin 15: dce-timing-plus
pin 16: dce-timing-minus
```

6. Whether a previously-generated waveform(s) is to be used (in which case a pointer to it is given) or new one(s) are to be obtained.

7. Whether the waveform(s) and analysis are to be displayed or not. Generally, they are displayed so that the operator may observe and understand the reasons for the diagnosis actions. However, when one TC contacts a remote TC to perform a measurement for it, waveform displays are generally suppressed at the remote TCF since presumably there is no one there to observe the actions.

#### 4.3.4.3 WAVEFORM SETUP

Before calling for waveform generation and analysis the diagnosis software "schedules" the scope, Mini-Matrix switch, and Datalok devices via calls to the MITEC Scheduler. Then the diagnosis process is able to perform the following actions without any possibility of access conflicts:

1. Tell the Telenex Mini-Matrix switch to (electronically) attach its monitor port to the location in the circuit that is to be measured. (The other end of the monitor port is attached to the scope.)

2. Tell the Datalok device to select the one or two pins on which signals are to be measured.

A waveform that the scope returns consists of 512 amplitude values each in the range 0 through 255. Somewhere in this range is the value corresponding to "zero" amplitude. Unfortunately, this "zero" amplitude is controlled by knobs on the scope and cannot be changed by external RS-232 commands. Therefore, before using the scope for retrieving and analyzing waveforms we must first determine this "zero" value via a sequence of commands referred to collectively as "calibrating" the scope. The basic steps in the calibration process involve telling the scope to connect its input to ground followed by reading in the resulting waveform. The mean of the amplitude values in that waveform is then assumed to be the "zero" amplitude. (We use the mean to average quantitative errors and to accommodate the possibility that the "zero" may not land on an integral value in the 0 to 255 range.) This calibration process is done for each channel and is then remembered for subsequent analysis of waveforms.

#### 4.3.4.4 WAVEFORM ANALYSIS

After MITEC has retrieved waveform(s) from the scope, it then analyzes them, with the ultimate goal being a pronouncement that the signal(s) in question are "good" or "bad".

The first step is to perform some basic arithmetic analysis in order to obtain the overall scope of the data, i.e., the positive and negative means, the minima, and the maxima. The scope's "zero" amplitude(s) obtained earlier during calibration, as described above, are used to obtain the zero crossings. The sweep rate and sensitivity values are used to scale the data appropriately.

Once the basic information has been extracted further analysis is performed based on the situation at hand and a table-driven set of specs for the kinds of signals that can be accommodated. A collection of pattern matching and analysis algorithms have been written for this analysis. These algorithms try to answer the following questions:

1. If synchronous, is clock rate within spec?
2. Is there a data signal?
3. If asynchronous, does signal return to mark voltage often enough?
4. If synchronous, do data transitions occur at the right time in clock cycle? (clock skew)
5. Are mark and space voltages within spec?
6. Are there instances of noise, glitches, or overshoots?
7. Is data transition rate within spec?
8. Are rise and fall times within spec?
9. Are maximum positive and negative excursions within spec?

The conclusions are returned in a hierarchically organized descriptive list. The top level of the hierarchy tells whether the waveform is good or bad while the inner levels provide increasingly specific information on aspects of the waveform.

For handling balanced signals the software extracts the signal and the common mode noise components of the readings and performs the analysis on the derived signal.

Other algorithms have been produced for comparing two clock signals and measuring the drift between them.

4.3.4.5 WAVEFORM DISPLAY

Analysis of waveform(s) does not require that the waveform(s) be displayed on the screen of the Symbolics terminal for the human operator to see. Such display, however, is very useful for development purposes, for demonstrations, and in general for a person to see and understand qualitatively what is happening. Therefore, the waveform(s) are generally displayed on the terminal. The waveform(s) are displayed so that the appearance is similar to what is visible on the scope. A background grid is provided, breaking the waveform area on the screen into a pattern of 10 boxes horizontal by 8 boxes vertical. If waveforms for two signals are displayed at once, one is displayed in the top area of the screen and the other in the bottom area. Typically, the data is on top

and the clock on the bottom if those are the signals chosen. The waveforms are labelled with information as to the signals involved and the values at the horizontal grid lines. (If only one waveform is displayed, it is centered in the waveform area on the terminal screen.)

To the right of the waveform display is an area reserved for general information about the waveform and a hierarchical analysis of its quality. The analysis provides a summary description of the results of the various tests that are performed on the waveform(s). See Figure 4.12 for a sample waveform.

#### 4.3.4.6 WAVEFORMS FOR LATER USE

The data relating to waveforms may be saved for later redisplay and use. During a diagnosis MITEC maintains a "history" of the relevant information about each step that is taken. Later on, the history of the diagnosis can be reviewed and studied.

### 4.3.5 SIGNAL QUALITY ASSESSMENT OF VF SIGNALS

MITEC determines via the database that the signal at a given access point is VF. A VF signal requires MITEC to use the VF monitor on the Telenex and an Hekimian 370x (where x equals 1, 3 or 5) test set. Expected signal levels and threshold values are maintained in the database so that MITEC can determine whether the signal quality is adequate or inadequate.

When we observe VF signal quality, we are using the Hekimian 370x to measure the signal level only. MITEC accesses the signal via the Telenex, tells the Hekimian whether to measure the transmit or receive direction and then awaits the level measurement. The Hekimian response contains two values, a level value and a frequency value. If the measured signal level is within the range of the value stored in the database then MITEC proceeds troubleshooting assuming the signal quality is good.

The Hekimian 370x is capable of much more than measuring the signal level. However, MITEC is currently concerned with only determining whether a VF signal is present or not. Other measurements pertaining to the VF signal quality might be called for given different circuit problems.

### 4.3.6 OBSERVATIONS

There are several observations we can make about the status of fault isolation in MITEC. Some of these issues are a comment on the current implementation status and are in our future plans. Other issues go beyond the current status and will need some type of resolution before MITEC can be deployed.

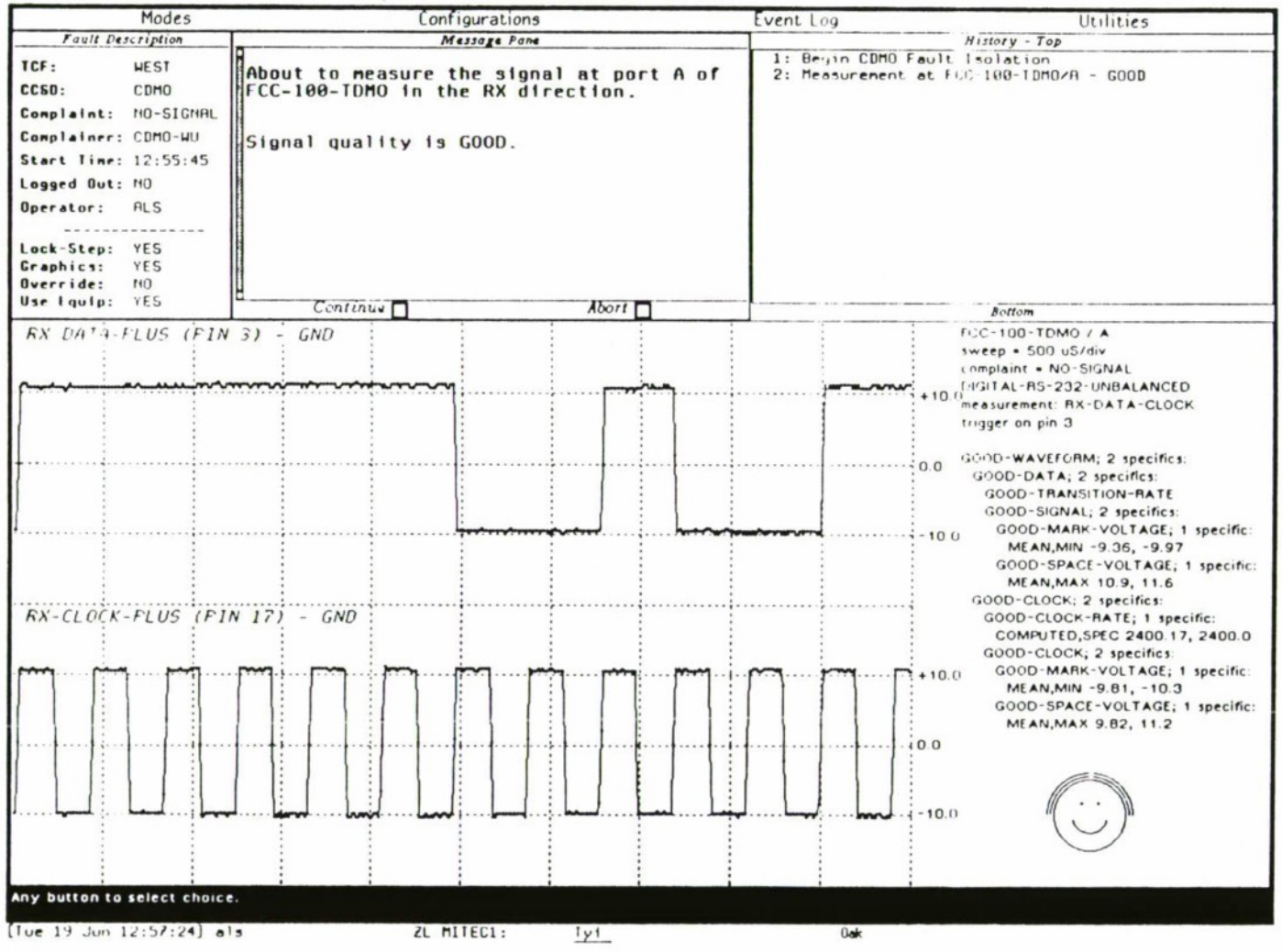



Figure 4.12
Waveform Display

MITEC troubleshoots loss-of-continuity problems on synchronous circuits. Based on the ETC experience, this type of circuit failure is thought to be a good framework to base diagnostic capabilities upon. Excessive of garbled transmission and too much noise are examples of additional complaints which should be handled in a deployable MITEC. Troubleshooting procedures for asynchronous circuits are also required.

Restoration procedures developed so far allow us to patch a circuit from one channel of a multiplexer to another channel on the same multiplexer. All the data representations involved are updated to reflect a patch though no changes are being written to disk. We plan to demonstrate patching in spare devices as well as circuit preemption. Another restoration scenario in progress shows how MITEC, given a failed T1 with a FCC-100 on a channel, can reconfigure the FCC-100 from 56Kbps to 9.6Kbps and reroute the FCC-100 trunk through a 9.6Kbps modem.

MITEC does not have any way to verify that the spare channel being considered for a reroute is indeed good. A deployed MITEC should have test equipment capable of generating known test tones for testing spare segments. Furthermore, it should be noted that we are not verifying that a patch was successful except for noticing that the users (pocket BERTs) are back in synch. This is comparable to a TC manually checking with a user that their communication services have been restored.

A critical issue that impacts MITEC is the air-gap, i.e., where MITEC is unable to perform some step electronically. Consider that during fault isolation some step requires manual intervention. If the operator tells MITEC an action was competed then MITEC will assume that the action really was completed and completed correctly. If, indeed, the action was not completed correctly then MITEC is likely to reach an incorrect conclusion.

# 5.0 TRAMCON/DPAS ALARM INTEGRATION

## 5.1 Introduction

A significant part of the FY89 effort in the MITEC project has been to begin to scope and understand the problems of correlating and interpreting transmission system alarms and presenting filtered results to Tech Control and Network Management decision makers. In particular, it is recognized that these decision makers will be able to function more rapidly and effectively if they can obtain immediate information about transmission system problems from TRAMCON and DPAS systems, rather than having to wait while the user community slowly reacts to transmission impairments and produces traffic pattern distortions that allow the managers to recognize and correct the problems. By analogy with the success of the Call-by-Call Simulator (CCSIM) as a portrayal of realistic network behavior to use in developing network management knowledge, it has been conjectured that TRAMCON and DPAS alarm generator systems could make up for the lack of access to real transmission networks for MITEC software developers, as well as the fact that real networks seldom produce interesting alarm patterns. An FY89 task for Lincoln Laboratory has been to define and specify these alarm generation systems, with an eye to beginning their implementation in FY90. Section 5.2 is the statement of required functions for the alarm generators that was written to serve as a guide for a study effort in the remainder of the year, and Appendix D is the result of that study, namely a Software Requirements Specification for the TRAMCON Event Generator (TEG).

## 5.2 Specification for TRAMCON and DPAS Alarm Generators

### 5.2.1 Background

Figure 5.1 illustrates 1) a military TRAMCON (TRAnsmission Monitoring and CONtrol) system, with the microwave radio network segment from which it gathers alarm information; 2) a DPAS (Digital Patch and Access System), for which the radio segment provides one or more of the T1 carriers to be switched and cross-connected; 3) a system control software facility, specifically the MITEC (Lincoln Laboratory Machine Intelligent Tech Controller) expert system, receiving and interpreting TRAMCON and DPAS alarm information; and 4) a simulation system that produces data streams realistically representing TRAMCON and DPAS alarm patterns, in terms of format, content, timing, and relationship to other network faults and phenomena being simulated elsewhere. (An example of such other simulation is the Call-by-Call Simulator of the Defense Switched Network, in which voice traffic behavior is affected by outages of trunk circuits carried by the microwave radio network.) The purpose of this Specification is to describe the requirements for the TRAMCON and DPAS alarm simulators.

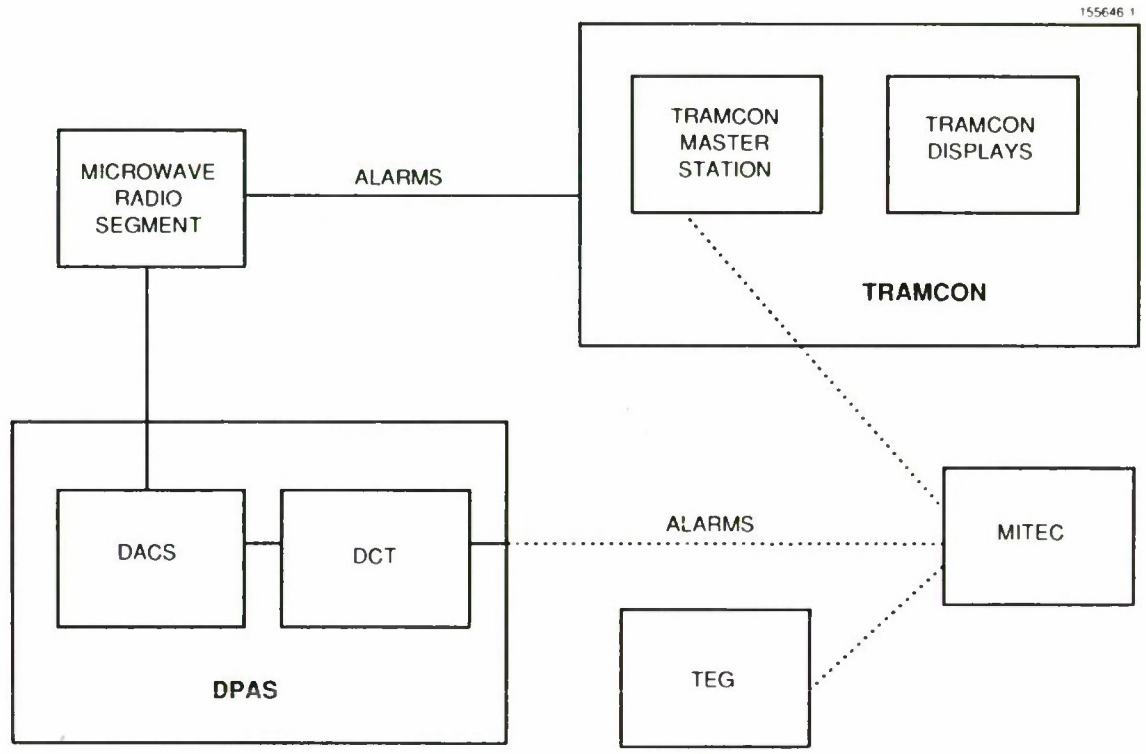



Figure 5.1
Alarm Monitoring Systems

### 5.2.2 Microwave Radio System Description

A Microwave Radio Segment consists of a string of microwave sites providing voice and data transmission via standard military transmitter/receiver equipment. Typically these sites also have multiplexing equipment which is used to combine lower-speed data streams from the local area into high-speed composite (i.e., T1 and T3) signals for radio transmission. All of this equipment is provided with various built-in fault detection features, with front-panel alarm lights coupled with electrical alarm signals that can be remotely monitored. There are moderately complex relationships among the various possible radio system failures, and the patterns of primary and sympathetic alarms they precipitate in other equipment, locally and at other sites. As one example, loss of transmitter power at an upstream site will cause a data loss alarm to occur at every downstream site.

### 5.2.3 TRAMCON Operation

There are two motivations for centralized monitoring of all the alarm signals generated by an entire radio segment: 1) some sites are manned only part-time, or not at all, and 2) simultaneous observation of alarm patterns from throughout a segment can lead to more rapid diagnosis of problems than would be possible from the restricted viewpoints of individual operators at local sites. These factors led to the development and deployment of TRAMCON systems, consisting of Datalok 10 alarm logging and command transmission devices at each radio site in a segment, all of which are connected to a TRAMCON Master Station located at a key site. The Datalok devices at all sites are polled on a regular basis by an HP1000 computer which is the main component of the TRAMCON Master Station; the alarm information is stored and summarized by the HP1000, and displayed in either color graphics or tabular form as selected by the operator.

A complete list of alarms monitored by TRAMCON is available elsewhere. Besides a variety of communications equipment alarms, this list includes critical facility information such as smoke detector outputs and emergency generator readiness. A skilled TRAMCON operator can recognize distinctive patterns of these alarms as presented by his display, quickly identifying the root causes of many different problems. In many cases the radio equipment will have switched automatically to spare modules, and the TRAMCON operator dispatches repair crews to correct specific problems in the on-line modules; in other cases the operator can alleviate problems by sending certain commands to the remote equipment via TRAMCON.

### 5.2.4 DPAS Description

Whereas TRAMCON is a deployed operational system, the procurement of DPAS systems is just beginning. Over 100 of them will

ultimately be deployed in the Defense Communications System. The DPAS consists of an AT&T DACS II (Digital Access and Cross-connect System, Version II) plus a DCT (DACS Control Terminal), together providing the capability to switch and multiplex numerous T1 signals at both the channel and subchannel levels. The DPAS provides its own set of alarm signals which are collected and displayed by the DCT, and follow standard commercial telecommunications practices as to their sources and meanings. Moderately complex relationships exist among microwave radio fault conditions and DACS alarms. An important objective of the work described here is to formulate and analyze these relationships.

### 5.2.5 Potential MITEC Interactions with TRAMCON and DPAS

The primary function of the Machine Intelligent TEch Controller (MITEC) expert system is to perform fault isolation and service restoral at a Tech Control Facility (TCF) for dedicated military circuits, including faults within the TCF as well as in transmission facilities linking it to other TCFs. In the latter case, human Tech Controllers can be very substantially aided in troubleshooting by having TRAMCON and DPAS alarm information available: they can pinpoint a trouble location more quickly by analyzing alarm data, rather than conducting a sequence of tests to infer the nature and source of the trouble. It is within the purview of MITEC to embody and apply the knowledge that human Tech Controllers use in exploiting TRAMCON and DPAS; the questions are:

1) how best to get the TRAMCON and DPAS information into MITEC, and

2) what reasoning and pattern recognition processes to use upon the information.

The operator displays at a TRAMCON master station contain the entire body of alarm information currently available. In principle MITEC could access the same data stream that drives the displays, but the useful information is enmeshed in screen formatting and control codes that are necessary for the color graphics monitor on the HP1000. One would not want to waste MITEC's resources in stripping out all of these unneeded codes; instead, the ideal course of action is ultimately to create a new interface module in the TRAMCON software that implements efficient computer-to-computer communication of the alarm information in a form that is directly exploitable by MITEC.

The fault alarm vocabulary of the DPAS is known, since it is identical to that of the DACS-II which is widely deployed in the commercial telecommunications environment. For T1 signals which are provided to a DPAS by way of military microwave links, the relationships among TRAMCON and DPAS alarms can be deduced and analyzed for a variety of fault conditions. In the future, when

DPAS systems are more widely deployed, simultaneous access to both DPAS and TRAMCON alarm information will be able to speed the diagnosis of various fault types. It is possible during the present ongoing development of the DACS Control Terminal to specify a computer-to-computer communication link to provide future access by MITEC to DPAS alarm data.

### 5.2.6 TRAMCON and DPAS Alarm Generator Requirements

The TRAMCON and DPAS alarm simulation system shown in Fig. 5.1 has two primary purposes: to support the development of MITEC software for alarm analysis and fault diagnosis, and to form a part of a broader simulation of Defense Communications System (DCS) operation. To meet these goals it must provide the following functions:

1. Storage of an internal representation of microwave and DPAS network segments, including all the equipment items and their alarm functions;

2. An operator interface which permits selection of any realistic failure event(s) in the networks;

3. A message interface which will permit failure event selection by a remote computer as an alternative to local selection by a human operator;

4. Internal generation of all the primary and sympathetic TRAMCON and DPAS alarms that would result from the selected failure event(s); and

5. Communication of this alarm information to MITEC in a manner consistent with the way TRAMCON and DPAS would communicate the alarms if they were provided with communication links to MITEC.

### 5.2.7 Development Sequence

It is clear that meeting all aspects of the requirements above would be a lengthy and expensive process. On the other hand, we believe that some modest level of completeness and precision, at a more modest cost in time and manpower, will satisfy our needs in the near term. We envision addressing this development in several stages, so as to identify and reach that modest level. Since we are breaking new ground in this effort, it does not make sense to try to tightly specify milestones, manpower and completion dates the way one could do for a job that closely resembled past projects.

The first step will be to develop a more complete software design specification. This will include analyzing a variety of information on the TRAMCON system, the radio segment alarms and

their relationships, and the DPAS system and alarms. Much of this information and analysis is already available at Lincoln Laboratory, and more will be obtained as necessary from a variety of sources (to be listed separately). This activity will lead to an initial understanding of the rules that govern the creation of alarm patterns. Preliminary ideas on the hardware/software environment for the alarm generation system will be advanced, and a high-level software design will be prepared. It is anticipated that this first step will be done by the end of FY89, with the level of completeness and sophistication to be determined adaptively as the work progresses.

The second step, to be undertaken in FY90, will be to choose an appropriate set of functions from this software design and proceed with initial implementation. Test and experimentation with the resulting system will lead to practical choices for further work. The goal is to have a practical, working TRAMCON/DPAS alarm simulator in hand in FY90.

## APPENDIX A

MITEC Equipment List

The following tables list the equipment in the two MITEC testbed systems (EAST and WEST) at Lincoln Laboratory. Estimated costs are $155,607 for EAST and $168,754 for WEST.

### MITEC EAST

| # | DESCRIPTION | SERIAL NUMBER | PROP# | PO# | COST |
|---|---|---|---|---|---|
| 1 | ADC PSM-16 Patch Panal | n/a | | | 1216 |
| 1 | ADC PSM-16 Patch Panal | n/a | | | 1216 |
| 4 | ADC PJ390A Telephone Jacks | n/a | | | 668 |
| 1 | EMUCOM Mocham 2400 | 1477 | 3-22479 | AX66440 | 876 |
| 1 | Digilog 620-I Protocol Analizer | 6908017 | 3-55260 | CX12064 | 12896 |
| 1 | Telecom Tech. Fireberd Error Analyzer | 5743 | 3-51873 | CX7387 | 11246 |
| 1 | Telenex Matrix Switch | 4210-1083 | 3-55191 | CX11335 | 20378 |
| 1 | IDS Pocket BERT 7262 | 42132 | 3-55009 | AX70152 | 550 |
| 1 | IDS Pocket BERT 7262 | 43814 | 3-55006 | AX65873 | 550 |
| 1 | PTT 5100 Telephone Line | 8810001 | 3-21793 | AX65423 | 6540 |
| 1 | Philips 3352/50 Digital Scope | 652 | 3-54488 | AX66532 | 4615 |
| 1 | Pulsecom Scanner | | | | 266 |
| 1 | Pulsecom Datalok 10A | 89-02 | 3-23243 | AX67179 | 1266 |
| 1 | Hekimian 3703 Remote Test Set | 3096 | | CX10036 | 4610 |
| 1 | Hayes Smartmodem 9600 | 1673339 | 3-21257 | AX64687 | 1599 |
| 1 | W+G Line Simulator | 0009 | 3-22928 | AX67500 | 5550 |
| 1 | Lambda Power Supply | n/a | 3-54862 | AX68677 | 1029 |
| 1 | Intraplex TDM-153 | 8810126 | | | 4400 |
| 1 | Intraplex TDM-153 | 8612151 | 3-52609 | AX63762 | 4400 |

| | | | | |
|---|---|---|---|---|
| Data Products AN/FCC 100 Simulator | 2657B | GFE288302 | | 10000 |
| ADC Mini DSX Patch Panal | | | | 302 |
| ADC Mini DSX Patch Panal | | | | 302 |
| HLI 3202 Line Relay Shelf | 2125 | 3-54995 | AX70102 | 3150 |
| HLI 3202 Line Relay Shelf | 669 | GFE290502 | | 3150 |
| HLI 3210 Access Control Shelf | 220 | GFE290503 | | 6570 |
| HLI 3280 AC Power Supply | 250 | GFE290501 | | 950 |
| HLI 3230 Testline Appearance Panal | 205 | GFE290505 | | 400 |
| Sun Micro Systems 3/260 Computer | 710E4715 | 3-12296 | BX1765 | 44000 |
| Equipto Electric 3 Frame | | 3-26010 | | 2250 |
| Cabinet 3 Pulizzi AC Power Controllers | | | | 586 |

MITEC WEST

| # | DESCRIPTION | SERIAL NUMBER | PROP# | PO# | COST |
|---|---|---|---|---|---|
| 1 | ADC PSM-16 Patch Panel | n/a | | AX-27404 | 1216 |
| 1 | ADC PSM-16 Patch Panel | n/a | | AX-27404 | 1216 |
| 4 | ADC PJ390A Telephone Jacks | n/a | | AX-71012 | 167 |
| 1 | EMUCOM Modem 2400 | 272 | | | 870 |
| 1 | Digilog 620-I Protocol Analizer | 6908016 | 3-55261 | CX12064 | 12896 |
| 1 | Telecom Tech. Fireberd Error Analizer | 5681 | 3-5-1872 | CX7387 | 11246 |
| 1 | Telenex Matrix Switch | 4210-1082 | 3-55188 | CX11335 | 20378 |
| 1 | Codex 26500 Modem | 6435 | 3-23232 | AX-67121 | 1165 |
| 1 | Codex 26500 Modem | 6426 | 3-23231 | AX-67121 | 1165 |
| 1 | Codex 26500 Modem | 6930 | 3-23859 | AX-68506 | 1050 |
| 1 | Codex 26500 Modem | 6948 | 3-23860 | AX-68506 | 1050 |
| 1 | Codex 26500 Modem | 7884 | | | 1050 |
| 1 | Codex 26500 Modem | 7882 | | | 1050 |
| 1 | Codex 26500 Modem | 189498 | | | 1050 |
| 4 | Codex Modem Shelf | n/a | | | 640 |
| 1 | IDS Pocket BERT 7262 | 42131 | 3-55008 | AX-70152 | 550 |
| 1 | IDS Pocket BERT 7262 | 43820 | 3-55007 | AX-65873 | 550 |
| 1 | P.T.T 5100 Phone Line Simulator | 8812007 | 3-22661 | AX66829 | 6540 |
| 1 | Domaine Sys. Remote Line Unit | 12510-510 | | L275093 | 225 |
| 1 | Domaine Sys. Remote Line Unit | 12524-524 | | L275093 | 225 |
| 1 | Data Probe 2wire/4wire Conv. | | | | 95 |
| 1 | Philips PM3352/50 Dig. Scope | 641 | 3-54308 | AX65188 | 4615 |
| 1 | Pulsecom Scanner | n/a | | | 260 |
| 1 | Pulsecom Dataloc 10A | 88-12 | 3-20733 | AX63410 | 1575 |
| 1 | Hekimian 375 PCM/VF Testset | 553 | 3-53883 | CX12025 | 11480 |
| 1 | Hayes Smart Modem 9600 | 1673291 | 3-21258 | AX64687 | 1599 |

MITEC WEST

| | | | | | |
|---|---|---|---|---|---|
| 1 | Lambda Power Supply | n/a | 3-54863 | AX68677 | 1029 |
| 1 | ADC Mini DSX Patch Panel | 824 | | L-273264 | 302 |
| 1 | ADC Mini DSX Patch Panel | 825 | | L-273264 | 302 |
| 1 | Intraplex TDM-153 | 8810127 | | AX71461 | 7570 |
| 1 | Data Products AN/FCC 100 | 2656B | GFE288301 | | 10000 |
| 1 | HLI 3202 Line Relay Shelf | 1692 | 3-54412 | AX63417 | 4040 |
| 1 | HLI 3202 Line Relay Shelf | 1692 | | CX10036 | 3420 |
| 1 | HLI 3210 Access Control Shelf | 350 | | CX10036 | 6570 |
| 1 | HLI 3280 AC Power Supply | 420 | | | 950 |
| 1 | HLI 3220 Control Panal | 212 | GFE90504 | | 3050 |
| 1 | Sun Micro Systems Computer 3/260 | 710E4715 | 3-12296 | BX1765 | 44000 |
| 1 | Equipto Electric 3 Frame Cabinet | n/a | 3-26011 | AX71013 | 2281 |
| | 3 Pulizzi AC Power Controllers | | | L273312 | 891 |

# APPENDIX B

## B.1.0 INTRODUCTION

A basic charter of the MITEC project is to develop techniques which minimize the human involvement in technical control and maximize analysis and decision-making by computer software. To achieve these goals, it is necessary for the computer to obtain status and alarm information directly (i.e., without human involvement) from communication devices; to issue commands to the devices to effect changes in status, connectivity, etc.; and to obtain measurements of signal strength and quality.

Section B.2 of this Appendix lists fourteen specifications for the required computer-device communication capabilities, using ASCII characters over an RS-232 line. Section B.3 provides motivation and elaboration on each of the fourteen specifications, and describes the consequences of lack of adherence to them. Section B.4 lists the specific devices with which the MITEC computer currently communicates.

## B.2.0 MITEC-Device Communication Specifications

This section lists fourteen specifications for computer-device communication that reflect experiences at Lincoln Laboratory in the development of MITEC. The specifications are concerned with the syntax (form) of the communication and do not address the semantics (content) of the messages that are exchanged. Since this section is written from the point of view of the computer, the specifications are oriented toward requirements to be satisfied by the devices selected for the project.

These specifications include "musts" and "desirables". Unless otherwise indicated, a specification is a "must" -- a basic requirement which must be satisfied in order for computer-device communication to be possible. Sometimes a device which violates some "must" requirements can in fact be accommodated by complex software or hardware work-arounds; it is best to avoid such situations. The "desirable" specifications are those that, while not absolutely necessary, could simplify software development; as such, they could affect the choice of devices for the project. The specifications are not ordered according to any scale.

### B.2.1 Command-Response Paradigm

Communication between a computer and the device shall consist of the following sequence:

1. The computer issues a "command" to the device to perform some action, change a state, supply some information, etc. The contents of legal commands are specified by the device and generated by software in the computer.

2. The device responds to the command with exactly one "response".

This command-response sequence shall continue as long as the computer issues commands.

B.2.2 Communication Character Set

All commands to the device and responses from the device shall be in printable ASCII characters, i.e., characters whose values are between octal 41 and 176 (inclusive) plus: tab (11), line-feed (12), carriage-return (15), and space (40).

Since the recipient of a response is a computer program, characters intended by the device designers for formatting the displays of a dedicated terminal are neither necessary nor desirable. Since display formatting characters generally involve non-printing ASCII characters their use was already precluded by the previous paragraph. This paragraph therefore covers the case of formatting characters that are printable, and serves to further emphasize the restriction.

B.2.3 XON/XOFF

The device and the computer shall each honor XON/XOFF flow control requests issued by the other. There is no requirement, however, that either party issue such requests.

An example of a situation in which a device would issue XON/XOFF requests is when the device is unable to accept an entire command together with all of its parameters at full input speed. An example where the computer would issue XON/XOFF requests is when it has asked for a dump of a waveform as a sequence of digital readings, and it is unable to meet the buffering requirements to accommodate the large response.

The requirement to support XON/XOFF flow control effectively prohibits transmission by the device or computer of 8-bit binary data since the XON and XOFF characters (control-Q and control-S, respectively) would be indistinguishable from the data.

B.2.4 Prompts

All responses from the device shall conclude with a unique string of one or more characters that does not appear in any other context. This string will be referred to as a "prompt".

B.2.5 Communication Path Initialization

It shall be possible to initialize the physical characteristics of the communication path (baud rate, parity, etc.) between the device and the computer once, during installation of the device. It shall not be necessary to re-initialize such characteristics after a power off-on sequence, crash, reboot, etc.

B.2.6 Establishing Known Device State

It shall be possible for the computer to issue a finite string of characters with the assurance that the device will then be in a "known" input state. The need for this occurs after events such as crash, boot, restart, or power off-on cycles of the computer or the device. It shall not be necessary for the computer to pace itself (by inserting delays) or to watch characters coming back from the device, to determine if the state has been reached.

B.2.7 Types of Responses

Each response from the device shall be one of the following:

1. Positive confirmation of receipt of the command;

2. Positive confirmation of receipt of the command plus the requested information; or

3. A message indicating the unacceptability of the command (an "error message").

B.2.8 Front Panel Commands

All commands to the device via its front panel shall be matched by equivalent commands which may be issued by the computer. (Excluded are actions involving changing external physical connections, power on, and power off.)

B.2.9 Asynchronous Device Output

Asynchronous output from the device (e.g., alarm messages) shall not be interleaved with responses to commands issued by the computer. Instead such asynchronous output shall be realized by one of the following two methods:

1. Transmission on a second RS-232 line from the device (with a suitable mechanism for delimiting the beginning and end of each such message), or

2. Production of a hardware alarm signal (e.g., a relay closure) which can be sensed by a device such as a DATALOK monitoring and control unit. If this alternative is chosen, then the device shall provide (synchronous) commands for use by the computer for obtaining specific information about the alarm.

B.2.10 Response After Action

In those cases where a command requests the device to perform an action or change a state (e.g., close a relay), the response shall be produced after the action is completed. The computer may therefore safely request subsequent other actions such as measurements, possibly from other devices, assured that there are no timing ambiguities.

B.2.11 Large Responses

Large responses shall be partitionable and a mechanism shall be provided for handling such large responses. (The word "large" shall refer to a mutually agreeable size, e.g, 1024, 1920 (80 * 24), others.)

Two acceptable mechanisms are:

1. The ability for the computer to request a selected portion of the output, e.g., mappings from x to y rather than all the mappings, or

2. A "more" mechanism in which an unambiguous and easily determinable indication is included in the response, indicating that the response is incomplete and that more information is available. If this alternative is chosen then the device shall provide a command which the computer may use for obtaining the next portion of the output. (This alternative is less desirable than the first one.)

B.2.12 Subcommands

Subcommands are neither necessary nor desirable for computer communication with the device. It shall be possible for the computer to issue any command together with all of its parameters without entering a "subcommand" mode in which each parameter is individually prompted for. The device shall be capable of receiving at full input speed (subject to XON/XOFF flow control) the entire command with all its parameters. (A subcommand mode for interactive human use is optional but not required.)

B.2.13 Echo (desirable)

The device shall support echo back to the computer of characters received by the device in commands from the computer, and shall support a command for turning the echo on or off. Character echo is useful when in a debugging or exploratory mode, and is unnecessary and undesirable when in an operational mode.

B.2.14 No Password Protection (desirable)

Password protection is neither necessary nor desirable. It shall be possible to run the device without having to provide a password.

B.3.0 Elaborations of the Specifications

This section is keyed to the fourteen specifications listed in Section B.2.0. It includes elaborations, motivations, experiences, and other insights into the specifications.

B.3.1 Command-Response Paradigm

This specification describes the basic style of communication between the MITEC computer and the devices; it is therefore a prerequisite for most of

the other specifications. With one exception all MITEC devices adhered to this specification. The exception was the FIREBERD 2000 Bit Error Rate Test Set which, under certain circumstances, gave two responses to one command: an immediate response followed by a delayed one containing the results of a test that was requested by the command. The dual response was circumvented by changing the command sent to the FIREBERD 2000.

B.3.2 Communication Character Set

Printable ASCII is desired because it simplifies the experimentation with the device during the exploratory phase: one can conveniently type commands to the device and read the responses using either a terminal or terminal-emulating software. Transmitting or receiving non-printable ASCII can be difficult or confusing in certain situations. Furthermore, this requirement makes it clear that one does not want the complexity of binary transmission, a problem that was faced with the DATALOK and was eventually circumvented via an octal virtual device (as described earlier). Finally, no real functionality is lost due to this restriction; printable ASCII is a rich enough communication medium.

The restriction against formatting characters in the text stream from the device is designed to avoid characters that a MITEC-like system does not need or want. Formatting characters waste bandwidth, need to be stripped out, and cause clutter in the software. If the MITEC implementers should decide to format information for human consumption or for reports, that task is their responsibility in the context of its requirements. MITEC simply wants the information so that it can make appropriate decisions.

B.3.3 XON/XOFF

Although honoring XON/XOFF flow control can be an issue for communication of any size, it is especially a problem in the case of large output of data from a device to the MITEC computer. Actual situations encountered to date include the following:

1. The transmission of a waveform from the digital scope to MITEC: This consists of 512 decimal integers (values between 0 and 255), plus 511 one-character separators between each two integers, plus header material, for a typical total of more than 2000 characters.

2. A dump of the DCE to DTE map (or its inverse) in the TELENEX mini-matrix switch: somewhat over 1800 characters. (This output would be substantially smaller were it not presented in tabular format, with many spaces inserted between the columns so that they will line up.)

Since the MITEC multiplexing computer (viz., a SUN workstation) is a general-purpose system designed to run a variety of software it is possible that situations may develop in which buffers for incoming characters may not be immediately available. Therefore, the operating system (UNIX) expects that devices communicating with it will obey the standard XON/XOFF flow control protocol. This is a normal expectation which is not dependent upon the particular choice of computer or underlying operating system.

the other specifications. With one exception all MITEC devices adhered to this specification. The exception was the FIREBERD 2000 Bit Error Rate Test Set which, under certain circumstances, gave two responses to one command: an immediate response followed by a delayed one containing the results of a test that was requested by the command. The dual response was circumvented by changing the command sent to the FIREBERD 2000.

### B.3.2 Communication Character Set

Printable ASCII is desired because it simplifies the experimentation with the device during the exploratory phase: one can conveniently type commands to the device and read the responses using either a terminal or terminal-emulating software. Transmitting or receiving non-printable ASCII can be difficult or confusing in certain situations. Furthermore, this requirement makes it clear that one does not want the complexity of binary transmission, a problem that was faced with the DATALOK and was eventually circumvented via an octal virtual device (as described earlier). Finally, no real functionality is lost due to this restriction; printable ASCII is a rich enough communication medium.

The restriction against formatting characters in the text stream from the device is designed to avoid characters that a MITEC-like system does not need or want. Formatting characters waste bandwidth, need to be stripped out, and cause clutter in the software. If the MITEC implementers should decide to format information for human consumption or for reports, that task is their responsibility in the context of its requirements. MITEC simply wants the information so that it can make appropriate decisions.

### B.3.3 XON/XOFF

Although honoring XON/XOFF flow control can be an issue for communication of any size, it is especially a problem in the case of large output of data from a device to the MITEC computer. Actual situations encountered to date include the following:

1. The transmission of a waveform from the digital scope to MITEC: This consists of 512 decimal integers (values between 0 and 255), plus 511 one-character separators between each two integers, plus header material, for a typical total of more than 2000 characters.

2. A dump of the DCE to DTE map (or its inverse) in the TELENEX mini-matrix switch: somewhat over 1800 characters. (This output would be substantially smaller were it not presented in tabular format, with many spaces inserted between the columns so that they will line up.)

Since the MITEC multiplexing computer (viz., a SUN workstation) is a general-purpose system designed to run a variety of software it is possible that situations may develop in which buffers for incoming characters may not be immediately available. Therefore, the operating system (UNIX) expects that devices communicating with it will honor the standard XON/XOFF flow control protocol. This is a normal expectation which is not dependent upon the particular choice of computer or underlying operating system.

Unfortunately, manufacturers who design devices for dedicated terminal usage do not always provide for XON/XOFF in the devices.

### B.3.4 Prompts

An end-of-response prompt is needed to inform the low-level communication software that a response is complete, and can therefore be delivered to the process in MITEC that generated the corresponding command. This permits reasonable modularization of communication software into low-level command-response handlers and higher-level processing of the responses. Moreover, in some situations (e.g., a variable-length response) the prompt is a crucial determinant of the end of the response.

A wide range of device behavior was encountered in this area. Designers of devices having dedicated terminals probably do not regard a prompt as necessary, since the person at the terminal can see when the response is complete. Some devices, e.g., the digital scope, do not provide prompts even though a dedicated terminal is not involved. Other devices provide prompts as specified above.

It is sometimes possible to interface successfully with a device that does not provide prompts. One solution is to structure commands in such a way that the response will contain a known sequence of characters at the end, and to assure that this sequence will not appear in any other context. This technique was used for the non-prompting digital scope. For commands to the scope that do not result in a response, a query command was appended in order to force a response. In such cases the semantic content of the response is ignored. For commands that do result in a desired response, such as a digitized waveform, the new-line character at the end of the response is treated as the prompt. As a side effect, the scope must be commanded not to use the new-line character elsewhere.

Another solution involves studying each response from the device, noting its concluding characters, and in effect treating each response as having its own prompt. To accommodate this kind of variable prompt, a mechanism was created in the low-level communication software for dynamically changing its record of the end-of-response prompt. This technique was used for handling subcommands which have prompts for parameters. A fatal flaw in this technique would occur for responses which conclude with variable information; so far, this fatal flaw situation has not been encountered.

For one device (the initial firmware version of the TELENEX mini-matrix switch) a novel solution to the absence of a prompt was devised. Whenever a command was sent to the switch, a timer process was commanded to wait n seconds and then issue a certain sequence of characters, the would-be prompt, in such a way that the low-level communication software would think that the device itself had issued the prompt. It was necessary to determine a reasonable number of seconds to wait, possibly modifying the wait-time to suit individual commands. Fortunately the current firmware version of the switch provides prompts, obviating the need for this complexity.

There are still problems, however, with the prompts produced by the current version of the TELENEX mini-matrix switch. The switch produces a prompt (viz., the ASCII ETX code) after it receives each character, not just at the end of the response. The switch also echoes numeric parameters that are sent to it in commands. Therefore, the response received from a typical command is a sequence of ETX characters possibly interspersed with digits. This was studied in detail for each command in order to determine the corresponding pattern. Then software was written to generate the appropriate sequence of characters that would be received from the switch for each type of command and to pass this sequence to the low-level communication software as the end-of-message prompt to be expected. Then the command is sent to the switch and a sequence comes back that matches the prompt exactly.

B.3.5 Communication Path Initialization

Invoking MITEC involves bringing up the software and readying the devices. It was not felt to be necessary to access each device (physically or via software) to place it in its expected physical operating state. Most devices remember transmission rate, parity, etc., across outages via either physical DIP switches or some kind of memory.

However, the digital scope does not remember its transmission rate across a power off-on cycle, reverting to 1200 bps after such a cycle. Since MITEC expects to communicate with the scope at 19,200 bps and since its rate is unknown when MITEC comes up, work-around software had to be implemented to change its rate from 1200 bps to 19,200 bps unless it is already at 19,200 bps.

B.3.6 Establishing Known Device State

The state of MITEC is the combined states of the MITEC computer and all the connected devices. When MITEC is running, the states of the devices are known at all times, permitting the MITEC computer to issue commands in the proper contexts. If the MITEC computer and/or devices go down and come back up (crash, boot, power off/on, etc.), the state information is lost. It is then essential to establish a known state for all the computers and devices in a quick and reliable manner.

The basic problem with establishing a known device state is that the MITEC computer does not know where the device was in its command interpretation software. For a device which has hierarchical command levels, the current level is important to know. If a command was being transmitted to the device at the time of the outage, it is important to know what the device last received. Without any of this knowledge, MITEC does not know what to send to the device.

A complex solution to this problem involves determining a sequence of characters which may be issued at any time (in the middle of a command or at new command input time), and will always result in a known device state. Some of these characters may cause error messages if they occur in certain states, but that is of no concern. The critical goal is that at the end of the sequence the device is in a known state. In order to determine such a

sequence one needs to study all the commands and all their states and then allow for all the worst cases. This is the approach used for most of the devices.

A better way of establishing a known device state is for the device to accept a specific reset character at any time. The current version of the TELENEX mini-matrix switch is the only device currently in MITEC that has such a character.

If the two methods described above for placing a device in a known state are not appropriate for a given device, then the ultimate fall-back position is to seize manual control of the device. Since MITEC is intended to be automated, this approach is clearly not desirable. To date, such manual intervention has not been found necessary.

B.3.7 Types of Responses

This specification requires that devices respond to commands and that responses fit into one of three categories. This may be regarded as a clarification of the entire command-response dialog rather than as an additional specification.

B.3.8 Front Panel Commands

It is generally inappropriate to allow manual operation of a device from its front panel at the same time that it is connected to MITEC in command-response mode. There may, however, be situations in which this is desirable. If so, it is essential that the MITEC computer be made aware of such intervention. Ideally the device would inform MITEC, via a response to a status request, that there had been human intervention. A less desirable way to inform MITEC is for the person to tell MITEC after the intervention is complete. Once again, a complete and accurate report is vital.

This specification does not preclude the inclusion in the MITEC environment of devices that can only be operated manually, as long as one realizes that such inclusion changes the basic nature of MITEC. Lack of adherence to this specification has not been encountered in any of the devices currently in MITEC.

B.3.9 Asynchronous Device Output

When a MITEC process issues a command to a device, the process is placed in an "awaiting response" mode. When the response finally arrives, it is returned to the command-issuing process, and the slate is cleared for the next such exchange. There is no mechanism for an unsolicited message from a device. Such an asynchronous device output would be regarded as a response without a prior command and would therefore be discarded. The devices currently in MITEC either do not generate asynchronous output, or they use the second method proposed in B.2.9, namely issuing a signal that is picked up by the DATALOK.

The first proposed method in B.2.9, a second RS-232 line, neatly fits into the command-response paradigm. A process in MITEC would issue a command to the "device" on the other end of this RS-232 line; a response would come back when an asynchronous output appears. The lengthy (perhaps infinite) delay between command and response would cause no problems in MITEC's multi-process design.

B.3.10 Response After Action

The MITEC computer must accept at face value the response from a device reporting that it has performed a requested action. The burden of determining that the action is complete should obviously be on the entity best equipped to know this, viz., the device.

B.3.11 Large Responses

MITEC buffers the characters of a response from a device until receipt of the prompt, at which point it returns the entire response to the process which issued the command. This specification attempts to place a limit on the size of the required buffer. To date the buffer size has been adjusted to 3000 characters to accommodate the largest possible response from the MITEC collection of devices. Total buffering is not an unreasonable burden at this point.

B.3.12 Subcommands

Subcommands featuring prompts for each parameter are intended for interactive human use. They get in the way during computer communication with the device. One approach that was tried with subcommands is to ignore the prompts for the specific parameters, and simply "type ahead" in one burst the sequence of characters that would be sent if one had paid attention to the prompts. Then one can treat the accumulated prompts for parameters as a single big response to be thrown away. While this approach works in many cases, there is one type of device design in which it fails. In that design the device either does not permit type-ahead or has a limited buffer for type-ahead. For such a device one must treat the combination of each parameter and the subsequent prompt for the next parameter as a command-response pair, with several such pairs needed to accomplish what is logically one command. This approach necessitates telling the low-level communication software to change the end-of-response prompt to track the changing responses (i.e., requests for next parameters) from the device.

B.3.13 Echo

This is an optional specification. There are three classes of devices: those which always echo, those which never echo, and those which adhere to this specification and permit turning echo on or off. All three types have been handled in MITEC. Of the two non-adherents one might prefer the class that always echoes. The wasted bandwidth due to the echoing is typically quite small, since the commands are typically quite short; the convenience of seeing what one is typing during experimentation is great.

### B.3.14 No Password Protection

While one might want to protect access to the MITEC computer via passwords, MITEC itself should not have to provide passwords to obtain access to the devices. If device passwords are required, then they are simply compiled into the MITEC software, rendering the entire protection system meaningless. The only device thus far encountered with password protection was the initial version of the TELENEX mini-matrix switch. In the current version, passwords are easily circumvented.

## B.4.0 Devices in the MITEC Testbed

The following is a list of devices in the MITEC environment with which commands and responses are communicated via RS-232 lines.

Philips Digital Storage Oscilloscope, Model PM3352
Telenex Mini-Matrix Switch
Datalok 10A with Relay Scanner
AN/FCC-100 Multiplexer
Hekimian 3200 Test Access Switch
Hekimian 3701, 3703, and 3705 Communication Test Sets
Fireberd 2000 Data Error Analyzer

# APPENDIX C

## C.0 MITEC-Device Communication Experiences

This Appendix provides general comments on our experiences with computer - device communication in MITEC.

### C.1 Vantage Points

It should be noted that interfacing MITEC with the various devices was approached from three different vantage points, depending on circumstances. While the requirements derived from the three vantage points overlap, there are enough differences that one needs to understand the varying requirements.

#### C.1.1 Experimentation

In this initial encounter the objective is to learn what it is like to communicate with the device.

#### C.1.2 Software Development

At this stage we have an understanding of the syntax and semantics of the messages to be sent to the device and the responses that can be expected in return. We are producing algorithms containing sequences of message exchanges with the device in order to make the device perform useful functions. We do not need verbose responses, help modes, or prompts for parameters, but would still like detailed error messages and echo of input.

#### C.1.3 Production Usage

Efficiency of use and reliable communication are the key factors at this stage. Echo of input is no longer needed, and we do not want lengthy acknowledgement messages. The aim is the minimal communication to accurately and reliably transact the operations.

### C.2 Devices Not Intended for the Purposes Used in MITEC

Many of the observed problems exist because the devices, for the most part, were not designed for operation in a MITEC-like environment (i.e., driven by software in an external computer). It is wrong to claim that since a device can handle RS-232 communication, it can therefore be easily driven by computer software.

The devices we encountered seemed intended for operation in one or more of the following environments.

#### C.2.1 Human Operation

Such devices are designed for operation by a person sitting at a terminal. Prompts are provided for parameters, help messages, and menus. Confirmation messages are issued for potentially dangerous commands. Output may contain

explanations to help the user understand its contents. Tabular output is frequently provided to ease the reading and searching of data.

The major problem in handling such devices is that this extra communication gets in the way when external software drives the device. The software must be programmed to deal with and skip the prompts for parameters, the help messages, and the often-lengthy menus. Confirmation requests must be automatically confirmed. Tabular output must be de-tabularized in order to extract the desired information.

A more serious set of problems may be dependence upon human recognition of implicit information. While a person can easily recognize implicit information, it may be difficult to produce software that can operate at this level. For example, consider the process of recognizing the end of a tabular response from a device that does not issue an end-of-response prompt. The person can quickly tell by looking at the response that it is complete. It may be much more difficult to produce software that can recognize the end of such a response. In a modularly-designed system (such as MITEC) it may be critical that one software module be able to recognize the end of a response in order to pass the response onto the next module for processing.

### C.2.2 Dedicated Terminal Operation

Such devices are designed for operation from dedicated terminals by a person. The comments and problems cited in the preceding section apply here, but there are several additional problems. Since the devices attempt to exploit the screen-formatting capabilities of the terminals, the output contains embedded formatting characters. The software developer must therefore strip away screen-formatting characters in order to process the useful information.

In the case of the TELENEX mini-matrix-switch (both in the first and the current version) we have had to circumvent its dedicated terminal operation. The problem concerns responses that require more than one print line. The switch uses the terminal's formatting characters to achieve placement of output onto subsequent print lines. We wanted to achieve this same functionality in our development and production environments via standard carriage-return/line-feed means. We programmed a special mechanism in our low-level communication software that would intercept the appropriate "new line" formatting characters and replace them with the expected carriage-return/line-feed sequence. The higher-level software is unaware that this transformation was performed and proceeds with the processing of the multi-line response.

Another issue in the context of dedicated terminals is graphical output. Such output involves drawings or schematics produced using the high resolution capabilities of a terminal or using appropriately chosen alphanumeric characters to approximate a drawing. As in the case of screen-formatting characters, such graphical output is of no use for MITEC. MITEC would need sophisticated, expensive code to decipher the output in order to extract the relevant information from such graphics.

## C.3 Severity of Problems

Problems encountered in communicating with devices can be characterized as to degree of severity and amenability to solution. At one end of the scale are those which are easily solved or circumvented via some straightforward software algorithms. An example is stripping out known and constant busy words from the device output stream in order to obtain the important information.

In the middle of the scale are problems which are difficult to solve and require some non-trivial software effort. An example is the DATALOK which communicates binary information (8-bit bytes) rather than ASCII characters. Keying in such 8-bit bytes from a terminal for experimentation can be awkward or impossible on some terminals. Additionally, understanding the output is difficult and time-consuming. To render the DATALOK useable we created in software a virtual octal device which communicates bi-directionally using only the characters 0 through 7 and comma. In the input or output stream a multi-digit octal number terminated by a comma represents the value of an 8-bit byte. The sequence of such numbers represents the communication with the DATALOK. Such communication is therefore very readable and understandable.

At the other end of the scale are problems which are fatal, or nearly so, to communication with MITEC. Short of removing the device from MITEC altogether, we must make some severe compromises and/or redesign to accommodate the device. Several examples are lack of support for XON/XOFF flow control, inability to place the device in a known device state, and the lack of end-of-response prompts.

# APPENDIX D TRAMCON Event Generator Design Requirements

## D.1. SCOPE

### D.1.1 Identification

This Software Requirements Specification establishes the requirements for the Transmission Monitoring and Control (TRAMCON) Event Generator (TEG).

## D.2 Purpose

### D.2.1 TRAMCON Mission

The TRAMCON alarm monitoring and reporting system collects and reports alarms associated with the Digital European Backbone (DEB). The DEB is a network of microwave relay sites that carries T1 trunks for the Defense Communications System (DCS) in Europe. Each TRAMCON system consists of one or more (normally one) TRAMCON Master (TM) stations and a number of remote units. The TM stations collect and process the data from the remote units which are connected directly to monitoring and control points on the transmission equipment. The remote units can be located at arbitrary distance from the central computer and communicate with the central computer via direct connection or remotely via modems.

The primary functions of TRAMCON are to remotely collect equipment status and performance data and to allow the remote operation of relay switches associated with the equipment. TRAMCON monitoring includes transmission alarms as well as facility alarms (e.g. power failure, intrusion, fire) and status. The TRAMCON systems also process the status and performance data in various ways to assist TRAMCON operators in assimilating and acting on the data. This processing typically includes the following:

a. Alarm processing, such as comparison of collected data to alarm threshold values.

b. Alarm correlation, i.e., the presentation to the operator of information on the status of related pieces of equipment.

### D.1.2.2 DPAS Mission

The Digital Patch and Access System (DPAS) provides for the control, monitoring, and patching of T1 trunk circuits (not only those carried by the DEB, but also other media). DPAS is implemented using the AT&T Digital Access and Cross-Connect System (DACS II) and has been projected to include a Network Control System (NCS) as well. DPAS and the DACS II equipment also monitor and report alarms associated with their components and the T1 trunk signals.

### D.1.2.3 MITEC Mission

The Machine Intelligent Technical Controller (MITEC) will assist the Technical Controller (TC) in troubleshooting telecommunications circuits in

a Technical Control Facility (TCF). MITEC will reduce TC manpower requirements and improve TC effectiveness by reducing the time to troubleshoot a circuit, automating currently manual quality assurance and recordkeeping activities, and refining the accuracy and consistency of troubleshooting and quality assurance.

D.1.2.4 Intelligent Alarm Filtering

The occurrence of a failure in the equipment monitored by TRAMCON and DPAS typically causes not only a primary alarm but also a number of sympathetic alarms (alarms that do not themselves indicate a fault but are consequences of the primary fault). When the number of these sympathetic alarms is large, it takes substantial skill and patience on the part of the human operator to identify the primary alarm.

Two systems under development at Lincoln Laboratory, MITEC and the Network Management Expert System (NMES), are intended to perform the task of identifying the fault that underlies a communications system failure and institute corrective action with a minimum of effort on the part of the human operator. MITEC addresses the diagnosis and rerouting of point-to-point user and trunk circuits; NMES addresses the diagnosis of switched networks and the application of network management controls. Both systems would profit from the introduction of TRAMCON and DPAS alarm data as a source of diagnostic information.

D.1.2.5 Role of the TRAMCON Event Generator (TEG)

In order to develop adequate techniques for the exploitation of alarm data in MITEC and NMES, a system is needed that can supply such data in response to a wide range of possible faults. TEG will be a simulator which produces the entire constellation of primary and sympathetic alarms associated with a fault and supplies these data to MITEC and NMES. The knowledge needed to define the relationships between faults and alarms already exists and has to some extent been recorded by USAF, Army, and DCS personnel.

The TRAMCON and DPAS alarm simulation system shown in Figure 1 has two primary purposes: to support the development of MITEC software for alarm analysis and fault diagnosis, and to form a part of a broader simulation of Defense Communications System (DCS) operation. To meet these goals it must provide the following functions:

a. Storage of an internal representation of microwave and DPAS network segments, including all the equipment items and their alarm functions;

b. An operator interface which permits selection of any failure event(s) in the networks which are recognizable by TRAMCON and/or DPAS;

c. A message interface which will permit failure event selection by a remote computer as an alternative to local selection by a human operator;

d. Internal generation of all the primary and sympathetic TRAMCON and DPAS alarms that would result from the selected failure event(s); and

e. Communication of this alarm information to MITEC in a manner consistent with the way TRAMCON and DPAS would communicate the alarms if they were provided with communication links.

The first increment of TEG will address TRAMCON events. The final delivery of TEG will address both TRAMCON and DPAS events.

D.2. APPLICABLE DOCUMENTS

D.2.1 Government Documents

ASISM 25-50-1, Information Management, DIGITAL SYSTEMS OPERATIONS MANUAL (DSOM) for DEB IIA, Support /Maintenance for DEB IIA Headquarters, U.S. Army Information Systems Command, Fort Huachuca, Arizona, September 1985.

CLIPS Reference Manual, Version 4.3 of CLIPS, Artificial Intelligence Section, Lyndon B. Johnson Space Center, June 1989.

Computer System Operator's Manual for the Transmission Monitor and Control System (TRAMCON), Version 1.8, Command and Control Systems Office, CCSO Information Systems Division, Tinker AFB, Oklahoma, 25 February 1988.

DEB Equipment Troubleshooting Procedures for Equipment Alarms, Det 12, 1945th C.G. Feldberg RRL, Germany Technical Control Operations, 29 June 1989.

DPAS and TRAMCON Interoperability Study, AT&T/Harris DCS System Integration Team, Arlington, Virginia, 05 November 1988.

Giarratano, Joseph C., Ph.D., CLIPS User's Guide, Version 4.3 of CLIPS, Lyndon B. Johnson Space Center, June 1989.

J4AMF/ASF/AST304X0 -033, PDS Code 9DC, Technical Training, TRAMCON-DATALOK 10, FTD 936, Rhein-Main AB, Germany.

Master TRAMCON Alarm Listing for TRAMCON Alarms, Det 12, 1945th C.G. Feldberg RRL, Germany Technical Control Operations, 29 June 1989.

Prime Item Development Specification for the Transmission Monitor and Control System (TRAMCON), CCSO/COI, Tinker AFB, Oklahoma, 05 February 1988.

TRAMCON Alarm Description for TRAMCON Alarms, Det 12, 1945th C.G. Feldberg RRL, Germany Technical Control Operations, 29 June 1989.

TRAMCON On-Line Software Reference Manual, June 1988.

TRAMCON Phase I Baseline, 17 June 1986.

Troubleshooting the D.E.B. Digital Equipment thru TRAMCON Alarms, Det 12,

1945th C.G. Feldberg RRL, Germany Technical Control Operations, 29 June 1989.

D.2.2 Non-Government Documents

365-301-002, AT&T, DACS II Reference Manual, AT&T, May 1987.

365-301-603, AT&T, DACS II Input/Output Message Reference Manual, AT&T, May 1987.

365-301-610, AT&T, DACS II Generic 2 Input/Output Message Manual, AT&T, Dec 1988.

Booch, Grady, Software Engineering with Ada, Second Edition, Benjamin / Cummings Publishing Company, Inc., Menlo Park, California, 1986.

D.3. REQUIREMENTS

D.3.1 Programming Requirements

The following subparagraphs establish the requirements for programming TEG. These subparagraphs establish the language and language processors to be used as well as guidelines to be followed in programming.

TEG is intended to be a knowledge-based system. That is, TEG will represent in declarative style the knowledge of equipment, circuit connections, fault trees, alarms, and polling needed to simulate TRAMCON alarms (Figure 2). This knowledge will be represented in the form of assertions that declare that certain facts and relationships exist, as opposed to the form of procedures that embody these facts and relationships in their execution.

TEG will employ a relational view of this encoded knowledge, rather than a more general object-oriented view (Figure 3). The relational view provides a means of representing data that is more directly useful in production rules than an object-oriented view would be; also, the relational representation would be more easily stored in and retrieved from a relational database management system (RDBMS), should this become necessary in the future.

TEG will be a rule-based system. That is, TEG will represent the procedures needed to manipulate data and produce alarms as "IF...THEN" rules, also known as production rules. However, unlike many rule-based systems, the particular facts and relationships known to TEG will be represented as assertions, while the rules will operate generally over the asserted data (Figure 4).

TEG will employ both data-driven ('forward chaining') and goal-directed ('backward chaining') formalisms. The majority of TEG execution will be data-driven. The goal-directed formalism will be employed to limit search and computation when this is necessary (Figure 5). Procedural programming will be used to implement external interfaces and to perform housekeeping.

D.3.1.1 Programming Languages

The various functions of TEG shall be programmed in languages suited to their requirements. Requirements and recommendations for each of these languages are given in this paragraph. Use of alternative languages will be permitted upon submission of adequate justification and approval of the Lincoln Laboratory Program Manager.

D.3.1.1.1 Rule-Based Language

All of the TEG simulation shall be programmed in a rule-based language. This language shall fulfill the following requirements:

a. Facts. There shall be a mechanism to assert and retract facts. There shall be a static class of facts that, once read in, remain present. There shall be a dynamic class of facts that may be freely asserted and retracted. There shall be a means to represent facts that is semantically equivalent to a relation.

b. Rules. There shall be a mechanism for the data-driven execution of rules. The rule language shall include a means for selection and binding of data used in rule execution. The rule language shall include a means to execute both procedures coded in the rule language and procedures coded in an external language. Note: If data-driven execution is provided, there is no need for an additional goal-directed execution mechanism. As discussed in 3.1.3 below, it is practical to emulate goal-directed execution under a data-driven mechanism.

c. Embedding. There shall be a means to embed the rule-based language in a program written in a conventional language and a means to call functions written in a conventional language from the rule-based language.

d. Efficiency. The rule-based language shall employ an algorithm to limit the amount of search needed to select data. This algorithm shall be at least as effective as the widely used Rete algorithm.

e. Portability. Programs written in the rule-based language shall be portable without modification of source code, so long as the target computer provides adequate resources. The minimum scope of portability shall encompass the IBM PC/AT (PC-DOS operating system), the Sun 3 (Berkeley Unix), and the AT&T 3B2 (System V Unix).

f. Multitasking. The rule-based language shall be capable of being executed in a multitasking environment.

The rule-based language for TEG should be CLIPS. CLIPS is deemed to meet all of the above requirements. Alternative rule-based languages may be used subject to justification and approval.

### Conventional Language

Those portions of TEG that are not practical to program in a rule-based language shall be programmed in a conventional high-order language (HOL). The following components of TEG are deemed suitable for programming in a conventional HOL:

a. System interface. This component includes procedures to execute and terminate execution of TEG, to interface to system resources required for TEG operation, and to perform housekeeping required for TEG operation.

b. Man-machine interface. This component includes procedures to format and present displays to TEG user and to accept and validate inputs from the user.

c. Database interface. If an RDBMS is used to store and retrieve facts, then the interface between the rule-based language and the RDBMS may be written in a conventional HOL.

d. Multitasking. The HOL shall be capable of being executed in a multitasking environment.

Additional components of TEG may be programmed in a conventional HOL, subject to justification and approval.

The conventional HOL for TEG should be Ada. For the initial implementation of TEG only, the C language may be used without additional justification or approval.

It is believed that assembly language will not be needed to accomplish any function of TEG. Use of assembly language would compromise portability of TEG. Assembly language shall not be used for any programming without justification and approval of the Program Manager. Justification for any use of assembly language shall show why it is impractical to accomplish the programming objective without the use of assembly language.

### Database Language

Should it prove necessary to store and retrieve facts from mass storage (as opposed to maintaining all facts in main memory), an RDBMS shall be used to implement the Database function. The RDBMS shall be capable of executing the following functions:

a. Projection. The RDBMS shall be capable of retrieving a table containing a set of selected attributes (a projection). It shall be possible to retrieve a projection that includes data from more than one table (a join). It shall be possible to specify commonly used projections (views).

b. Restriction. The RDBMS shall be capable of retrieving only those data that satisfy a given predicate (a restriction). The RDBMS shall support compound predicates and shall recognize, at a minimum, the following operators: the logical operators AND, OR, and NOT; the numerical relation operators <, <=, =, |=, >=, and >; the arithmetic operators +, -, *, /, and mod; and the relation operators = and |= for strings.

c. Transaction processing. The RDBMS shall be capable of executing a sequence of operations as a transaction. It shall be capable of updating the database on transaction completion ('commit') and of restoring the database to the pre-transaction state on transaction abort ('rollback'). It shall be capable of establishing checkpoints and of restoring the database to the checkpointed state. An archive/restore mechanism is an acceptable means of implementing the checkpointing requirement.

d. Interface. The RDBMS shall provide a means of interfacing to the selected conventional HOL. This interface should consist of RDBMS functions callable from the conventional HOL.

The RDBMS shall use the language Structured Query Language (SQL). The RDBMS should be one of the following: ORACLE or UNIFY. Another RDBMS may be substituted, subject to justification and approval.

D.3.1.2 Language Processors

Language processors chosen for TEG shall conform to the requirements of 3.1.1 and its subparagraphs above. The language processors shown in Table 1 are recommended; these are deemed to satisfy the requirements of 3.1.1. Alternative language processors may be substituted, subject to justification and approval.

TEG shall be programmed to compile and operate correctly under the language processor versions current at the time of delivery.

Table 1
Language Processors Suitable for TEG

| Computer/Operating System | | | | |
|---|---|---|---|---|
| Target | IBM PC/PC-DOS | Macintosh | Sun 3/Unix | AT&T 3B2/UNIX |
| Rule Lang<br>Conv. HOL | CLIPS/PC | CLIPS/Mac | CLIPS/Unix | CLIPS/Unix |
| Ada | TBD Ada | Ada Vanatage | Verdix Ada<br>Ready Systems RTAda | Verdix Ada |
| C | Microsoft C<br>Turbo C | THINK C | Unix C | Unix C |
| RDBMS | ORACLE<br>XDB-SQL | ORACLE | UNIFY<br>ORACLE | UNIFY |

D.3.1.3 Programming Standards

The programming standards herein established are intended to assure that TEG will conform to the programming model discussed in 3.1 and that TEG will meet its objectives for understandability, maintainability, expandability, reusability, and portability. The following programming standards apply to the programming of TEG:

a. Rule programming style.

i. Programming in the rule language shall employ forward chaining to the extent practical, backward chaining when necessary as an alternative to forward chaining, and procedural code as a last resort.

ii. Use of control facts should be minimized. Generally, control facts that represent goals and subgoals in a backward chaining formalism will be permitted, while other types of control fact should be carefully justified.

iii. Use of salience should be minimized.

iv. If the rule-based language supports deterministic order of rule firing based on the sequence in which the rules are loaded, rule loading order may be used to control execution. This method of controlling execution order should only be used as an alternative to either salience or adding one or more additional rules whose sole purpose is to control the order of execution. Where rule loading order is critical, the concerned rules must be contiguous within the same file; furthermore, a comment shall be provided for

each of the concerned rules stating the required sequencing relationship among these rules.

v. If CLIPS is used for the rule language, the guidelines to programming style in the CLIPS User's Guide should be followed.

b. Conventional HOL programming style. Programming in the conventional HOL shall be in accordance with the commercial practice commonly known as 'structured programming'. Software Engineering with Ada should be used as a guide.

c. System-dependent code. The writing of system-dependent code is discouraged unless it is impractical to accomplish a desired function in a system-independent fashion.

i. If it is necessary to write code that is dependent on particular characteristics of a language processor, operating system, host computer, or peripheral equipment, then this code shall be identified and isolated. Information hiding shall be used to restrict the scope of impact of system-dependent code.

ii. If conditional compilation is available in the language processor(s) used, then this shall be used to ensure that the system-dependent code is compiled only for those systems that require it. In this situation, it is possible (and permissible) that several versions of code specific to particular systems will exist and that only one version will be compiled.

iii. When alternative versions of system-dependent code exist, these shall be grouped together.

iv. It shall be acceptable to use the Unix termcap facility as a mechanism for avoidance of system-dependent programming.

d. Control of recompilation. Recompilation of those software components that require it (because dependent on a changed component) shall be accomplished automatically (as in Ada or Turbo C) or through a makefile (as in Unix C).

e. Self-modifying code. No self-modifying code shall exist in TEG. In the rule-based language, rules shall be considered to be code; facts shall not.

f. Table names. Duplication of table names (e.g. deffacts blocks in CLIPS) shall only be allowable between tables when the intention is to allow one of the tables to override the other.

g. Modularity. Each functional group of rules, data or procedures shall be contained in a separate file. There shall be a header section for file. The format of these header sections shall be consistent for all such files of the same type, e.g., all rule file headers shall conform to the same style.

h. Readability.

i. All variable names shall be meaningful.

ii. Each rule and procedure shall have a comment explaining its function. Any restrictions, or assumptions shall be explained.

iii. If two or more rules are similar, a comment should be provided with each of these similar rules which distinguishes the purpose of each rule.

iv. In the case of CLIPS, the wildcard (? or $?) should be used, rather than variable names, to represent items which are don't care in rule patterns. The reason for this is to allow the reader to readily identify the fields that are pertinent to the matching process.

i. Performance.

i. Any unnecessary clauses inserted for the future or 'just in case' shall be commented out until the need is identified.

ii. The use of multifield variables shall be minimized. This type of construct shall be permitted only when the number of items cannot be predicted in advance or failure to use this construct will result in the production of additional rules.

iii. Facts should be cleaned up as soon as they are no longer appropriate.

iv. Statements whose sole purpose is to test variable values against constants should be avoided if the test(s) can be accomplished within the pattern matching statements (e.g. use of "~" or "&:" construct in CLIPS).

j. Maintainability.

i. Automatic symbol generation facilities (e.g. the gensym function in CLIPS) shall be used, at most, sparingly.

D.3.2 Design Requirements

D.3.2.1 Sizing and Timing Requirements

The performance of TEG shall be validated against the following timing criteria while operating, on a 3B2 600 or equivalent machine:

a. A response to a request from MITEC shall be ready for transmission to MITEC within forty (40) seconds of receipt from MITEC.

b. Each packet of information shall be transmitted to MITEC within one (1) second of receipt of acknowledgement of the previous packet.

c. A response to a request from the operator shall be presented to the operator within two (2) seconds of receipt.

d. Program start and re-initialization shall be completed within one (1) minute.

TEG shall be capable of running on a 3B2 600 or equivalent machine equipped with a minimum of eight (8) megabytes of main memory. The disk storage requirement currently estimated is five (5) megabytes.

D.3.2.2 Design Standards

Software which is to be implemented using the high-order language shall be developed using structured methodology. Critical design decisions shall be validated by the use of prototyping prior to initiation of software coding activities. Prototyping shall be used to validate the man-machine interface and the feasibility of attaining system sizing and timing requirements.

D.3.2.3 Design Constraints

Fault propagation shall be performed with the use of a forward-chaining rule-based design.

D.3.3 Interface Requirements

D.3.3.1 Interface Relationships

This paragraph discusses the interface relationships between TEG and its interfaces. Figure 6 shows the TEG context diagram. It depicts the major data flows between TEG and its interfaces.

D.3.3.2 Interface Identification

This paragraph specifies the proper identification of each interface:

a. MITEC Interface

b. User Interface

D.3.3.3 Detailed Interface Requirements

TEG Interface to MITEC CSCI

MITEC is an ongoing government-sponsored Air Force project at M.I.T. Lincoln Laboratory to research and develop intelligent computer-controlled support to the area of technical control of government communication facilities. A basic charter of the project is to develop techniques which minimize the human involvement in technical control and maximize the analysis and decision-making by computer software. As such, it is necessary for the computer to obtain directly, i.e., without human involvement, status and alarm information forum communication devices; to insure commands to the devices to effect changes in status, connectivity, etc.; and to obtain

measurements of signal strength and quality.

This Section attempts to provide specifications for such computer-to-device communication using ASCII characters over an RS-232 line. It is concerned with the 'syntax' (form) of the communication and does not address the 'semantics' (content) from the point of view of MITEC.

### Command - Response Paradigm

Communication between MITEC and TEG shall consist of the following sequence:

a. MITEC issues a 'command' to TEG to supply information. The contents of legal commands shall be specified by TEG and will be generated by MITEC software.

b. TEG responds to the command with exactly one 'response'.

This command-response sequence shall continue as long as MITEC issues commands.

### Communication Character Set

All commands from MITEC and responses from TEG shall be within the set of printable ASCII characters, i.e., characters whose values are between octal 41 and 176 (inclusive) plus: tab (11), line-feed (12), carriage-return (15), and space (40).

Characters intended for formatting a terminal display shall not be included in responses to MITEC.

### XON-XOFF

TEG shall be capable of accepting any XON-XOFF flow control requests issued by MITEC.

### Prompts

All responses from TEG to MITEC shall conclude with a unique string of one or more characters that does not appear in any other context. This string will be referred to as a 'prompt'.

### Communication Path Initialization

It shall be possible to initialize the physical characteristics of the communication path (baud-rate, parity, etc.) between TEG and MITEC once during installation of TEG. It shall not be necessary to re-initialize such characteristics after a power off-on sequence, crash, reboot, etc.

### Establishing Known Device State

TEG shall be capable of accepting a synchronization string from MITEC. This synchronization string will consist of a finite string of characters. The

following types of synchronization strings shall be supported:

a. re-initialize TEG

b. suspend simulation until receipt of next command from MITEC

It shall not be necessary for MITEC to pace itself (e.g. by inserting delays) or to watch characters coming back from TEG to determine if the state has been reached.

Types of Responses

Every response from TEG to MITEC shall be one of the following:

a. Positive confirmation of receipt of the command.

b. Positive confirmation of receipt of the command plus the requested information.

c. A message indicating the unacceptability of any command from MITEC. Unacceptable commands shall be defined as including invalid commands, valid commands which are not valid in the current context, and commands containing transmission errors.

Asynchronous Output from TEG

There shall be no asynchronous output from TEG to MITEC; that is, output not in response to a command from MITEC.

Response After Action

In those cases where a command from MITEC requests TEG to perform an action or change a state, the response shall be produced after the action has been completed.

Large Responses

Large responses shall be partitionable and a mechanism shall be provided for handling such large response. A large response is one in excess of 2048 bytes or 24 lines, including the 'prompt'. TEG shall be capable of accepting requests for a selected portion of the output from MITEC.

a. The initial increment of TEG shall support a 'more' mechanism in which an unambiguous and easily determinable indication is included in the response to show that the response is incomplete and that more information is available. TEG shall provide a command which MITEC may use for obtaining the next portion of the output.

b. If returning alarms by category becomes useful in the future, the ability for MITEC to request a selected category of alarms (e.g., major equipment alarms) will be added.

Subcommands

It shall be possible for MITEC to issue any command together with all of its parameters without entering a 'subcommand' mode in which each parameter is individually prompted for. TEG shall be capable of receiving any command from MITEC with all of its parameters at full input speed (subject to XON/XOFF flow control).

Echo

Future increments of TEG may support a command for turning on or off the echo of input characters that appear in commands from MITEC. The default mode for this characteristic shall be 'echo off'.

No Password Protection

Entry of a password shall not be required in order to execute TEG.

TEG User Interface

The first increment of TEG shall support a user-friendly menu-driven (keyboard) interface. Future increments of TEG may additionally support a scripting interface that allows for batch processing.

The keyboard user interface shall not require the user to learn formatting and syntax rules. For the first increment of TEG, the User Interface to TEG may be a set of menus driven by the rule-based language. If the first phase is developed using CLIPS, the user interface inputs shall include a batch file for the TRAMCON segment that is modelled in the TEG database. This batch file shall include commands for clearing memory, loading all rules and static facts associated with the TRAMCON segment, and performing the commands that cause a re-initialization of the rule-based environment (e.g., CLIPS reset).

Menus that allow the operator to select from a set of various functions will provide a consistent set of options with a visual cue to allow the operator to differentiate between available and unavailable options based on the current state of the simulator.

Selection of any function which requires selection of additional criteria will result in the presentation of sub-menus allowing the user to either select from the set of applicable values or respond to a prompt with a brief textual response. For a particular function, these menus and prompts will be presented in a consistent sequence. When a particular sub-menu or prompt is not applicable based on previous selections, that sub-menu or prompt will not be presented.

All user responses shall be validated. Selection of an unavailable option shall be considered an invalid user response. An invalid user response shall never cause the simulation to terminate prematurely or otherwise behave in an abnormal fashion. Because the required user responses shall be brief and simple, and the set of valid user responses and valid formats shall always

be clearly stated on the menu or prompt, a user shall always be given an indefinite amount of attempts to re-try after entering an invalid response.

An escape mechanism will be provided to allow the user to cancel a function selection prior to completing selection from all sub-menus associated with that function. When this escape mechanism is invoked, control will be returned to the menu from which the function was originally selected.

The user interface associated with future increments of TEG shall provide a means for selection of TRAMCON segments, and the loading and saving of scenario files.

TEG Interface to MITEC Network HWCI

TEG will be connected to MITEC using an RS-232 connection.

## 3.4 Functional and Performance Requirements

TEG shall simulate a single TRAMCON segment at a time.

### 3.4.1 Simulation Input Function

The Simulation Input function shall be responsible for receiving and pre-processing all inputs to TEG.

Simulation Input Function Inputs

Inputs to the Simulation Input function shall include the following:

a. Keyboard inputs from the operator

b. TEG knowledge base

c. Commands from scenario files

d. Messages from external programs and/or devices

Simulation Input Function Processing

The keyboard interface shall consist of a hierarchy of menus displayed appropriately for obtaining direction from the operator. Where appropriate, text, rather than a menu selection, is the expected response from the user.

The main simulation functional menu shall include, at a minimum, the following options:

a. modify polling sequence

b. change automatic switchover state

c. change operational side

d. insert a fault

e. remove a fault

f. poll

g. run simulation

h. exit simulation

Each of the above options, except for the exit simulation option, shall allow the user to specify an associated time. Time will be accepted from the user in hh:mm:ss format. A default time shall also be provided by TEG. An example of default time is the last time entered by the user.

Selection of the modify polling sequence option shall cause the current polling sequence to be displayed and allow the user to specify the exact order of polling of the TRAMCON sites within the selected TRAMCON segment. The user shall have the capability to specify any permutation of these sites. Any site(s) may be omitted from the polling sequence. For example, if the available sites are BST, HST, and RAG, any of the following sequences may be specified:

| | | | |
|---|---|---|---|
| BST, HST, RAG | BST, RAG, HST | HST, BST, RAG | HST, RAG, BST |
| RAG, HST, BST | RAG, BST, HST | BST, HST | BST, RAG |
| HST, RAG | HST, BST | RAG, HST | RAG, BST |
| RAG HST | BST | none | |

After a complete valid user response is entered, it will be forwarded to the applicable function(s) for further processing.

Selection of the change auto switchover state option shall cause a menu of sites to be displayed. After a valid user response is entered, a menu of equipment located at the selected site that has more than one redundant side will be displayed. After a valid user response is entered, a menu of valid switchover states will be displayed. After a valid user response is entered, the database shall be updated to reflect the new auto switchover state.

Selection of the change operational side option shall cause a menu of sites to be displayed. After a valid user response is entered, a menu of equipment located at the selected site that has more than one redundant side will be displayed. After a valid user response is entered, a menu of all sides applicable to this piece of equipment will be displayed. After a valid user response is entered, a primary fault event will be created for processing by the Event Generation function (3.4.2).

Selection of the insert fault option shall cause a menu of sites to be displayed. After a valid user response is entered, a menu of equipment located at the selected site will be displayed. After a valid user response is entered, a menu of primary faults applicable to the selected equipment will be displayed. After a valid user response is entered, a menu of ports applicable to the selected equipment will be displayed. After a valid user

response is entered, a menu of sides will be displayed if the selected equipment has redundant sides. After a valid user response is entered, the complete request will be forwarded to the applicable function(s) for further processing.

Selection of the remove fault option shall allow the user to select a fault for removal from the set of all faults that were directly inserted by the operator and have not yet been removed by the operator. The user shall be required to provide a repair time.

Selection of the poll option shall allow the user to specify one of the following types of polling:

a. Poll once at an absolute simulation time

b. Poll once at a relative simulation time

c. Poll n times starting at an absolute or relative simulation time at a specified or default polling frequency

d. Poll indefinitely starting at an absolute or relative simulation time at a specified or default polling frequency

e. Stop indefinite polling at an absolute or relative simulation time

Selection of the run simulation option shall cause the simulation to run for the specified period of simulation time.

Selection of the exit simulation option shall cause return of control to the operating system.

Selection of any of the above options, except exit simulation, shall cause the simulation menu to be redisplayed after completion of execution of the selected option.

In addition to the above options, the following functions will also be supported from the keyboard interface:

a. select TRAMCON segment

b. load scenario file

c. save scenario file

Selection of the select TRAMCON segment option will cause a menu of segments available within the TEG Knowledge Base to be displayed. After a valid user response is entered, the current segment will be set to the selected segment.

Selection of the load scenario file option will cause a list of all available scenario files to be presented to the user. The user will be capable of selecting a file (or no files) from this list. If a file is selected, the dynamic facts within that file will be loaded.

Selection of the save scenario file option will cause the save state to be toggled. The initial save state will be off. When the state is toggled on, the user will be prompted to specify a file name for the newly created scenario file. All selections made by the user between the time that the state is toggled on and the time that the state is toggled off will be saved within that file.

Once all required user inputs have been collected for a given option, the assembled inputs shall be validated to ensure that the entire request is valid. Valid requests shall then be forwarded to the appropriate function(s) for further processing.

The scripting interface will support the full functionality provided by the keyboard interface.

Messages from MITEC shall be validated and processed as if they had been entered by the user; except, in the case of an error in a message from MITEC, the simulator shall output an error message to MITEC and prepare to receive another message. The simulator shall not output menus to MITEC.

Simulation Input Function Outputs

The outputs of the Simulation Input function shall include:

a. menus

b. prompts

c. validated simulation requests

d. error messages to MITEC

e. message acknowledgements to MITEC

D.3.4.2 Event Generation Function

TEG shall support the following types of events:

a. equipment state transitions, resulting from primary faults or sympathetic faults

b. primary faults, resulting only from the insert fault option (3.4.1.2)

c. sympathetic faults, resulting from primary or other sympathetic faults
d. alarms, resulting from primary faults, sympathetic faults, or other alarms

Each possible alarm and sympathetic fault will have exactly one of the following causes:

a. the alarm/sympathetic fault can be caused by a single alarm/fault

b. the alarm/sympathetic fault can be caused by the presence of one or more of a set of causing alarms/faults (i.e., an OR condition)

c. the alarm/sympathetic fault can be caused by a combination of faults/alarms, all of which must exist for the alarm/sympathetic fault to exist (i.e., an AND condition)

Removal of an alarm/fault shall result in the following:

a. removal of all alarms/sympathetic faults caused by this alarm/fault alone

b. removal of all alarms/sympathetic faults which can be caused by this alarm/fault or other alarms/faults (OR condition), when no other causing alarms/faults exist

c. removal of all alarms/sympathetic faults which can only be caused by this alarm/fault in conjunction with other alarms/faults (AND condition), regardless of whether or not any such other alarms/faults exist

If the operational side of the applicable device is faulted and the non-operational side is not faulted, modification of the switchover status enabling automatic switchover results in switchover of the device to the other side.

If the non-operational side of the applicable device is not faulted, a manual switchover request results in switchover of the device to the other side; otherwise, manual switchover requests are ignored.

Either automatic or manual switchover of a device results in removal of any alarms/faults uniquely associated with the previous operational side.

Event Generation Function Inputs

The inputs to the Event Generation function shall include operator requests to add or remove events as well as the Event, Causality, Connectivity, Equipment, Equipment Status, Port, and Side tables. The fault constellation is also an input to the Event Generation function.

Event Generation Function Processing

This function shall produce and retract sympathetic faults, alarms, and equipment status. A sympathetic fault is a fault that is caused by a primary fault (a fault input by the user through the Simulation Input function, 3.4.1) or another sympathetic fault. When a fault (or occasionally a logical combination of faults) that is monitored at the Datalok device or TRAMCON occurs, an alarm results. Equipment status consists of various indicators that are monitored but are not considered alarms because they may arise in normal operation; equipment status may be input by the operator or change in response to the occurrence of faults.

Fault Propagation

The Event Generation function shall produce the entire constellation of sympathetic faults for one or more given primary faults. The process by which sympathetic faults are produced is called fault propagation. The Event Generation function shall carry out fault propagation according to the causality relationships in the CAUSES table (3.4.5.3.2.4.2). Fault causality relationships exist within equipment (CAUSES SAME in the CAUSES table) and between items of equipment (CAUSES SENDER, DISTANT, or LINK-END). The Event Generation function shall propagate faults for all of these causality relationships:

a. Fault propagation within a device (SAME relationships) shall occur whether or not the device (or the side of the fault in a device with redundancy) is on-line. Fault propagation between devices shall occur only between on-line devices (or between on-line sides in devices with redundancy). On-line status shall be determined from the Equipment Status (EQSTATUS, 3.4.5.3.1.4) table.

b. Fault propagation from sender to receiver (SENDER relationships) shall occur when there is a direct connection from the sender to the receiver. The presence of a direct connection shall be determined from the Connection (CONN, 3.4.5.3.2.1) table.

c. Fault propagation to the distant end (DISTANT relationships) shall occur when there is a connection (which may be direct or indirect) from a sender to its distant-end counterpart receiver. Connection tracing (3.4.2.2.2) shall be used to determine the distant end equipment.

d. Fault propagation to the opposite link end (LINK-END relationships) shall occur when the sending equipment and receiving equipment are the link ends of a circuit. The opposite link-end shall be determined from the Circuit (CKT, 3.4.5.3.2.3.1) table.

Fault propagation within an item of equipment is complex and allows for many combinations. The Event Generation function shall propagate faults for all of the following classes of causality relationship:

a. Within a port.

b. From a near port to the far port.

c. From the far port to all near ports.
d. To all ports on the equipment.

The Event Generation function shall support OR-causality of faults. In OR-causality, a fault shall occur when at least one of the causes of the fault has occurred.

The Event Generation function shall time-stamp faults. The time of a fault shall be the time at which the cause of the fault occurred. If the fault

has more than one cause, the time shall be the time of the first cause to occur.

Connection Tracing

When there is a fault on a device that causes a fault on its distant-end counterpart, connection tracing shall be used to determine the distant-end counterpart device. Connection tracing shall proceed from the location of the fault in the direction of signal travel until the corresponding port on a device of the same class has been reached. The direction of signal travel shall be assumed as follows:

a. For a fault originating on a near port, first to the far port of the same device and thence along the connection of the far port.

b. For a fault originating on a far port, along the connection of the far port.

The corresponding port on the distant-end device shall be determined as follows:

a. For a fault originating on a near port, any near port (not necessarily the same near port) of the first device of the same class encountered.

b. For a fault originating on a far port, the far port of the first device of the same class encountered.

Alarm Generation

Alarms shall be generated in the same manner as fault propagation: when a primary fault, sympathetic fault, or other alarm that is the cause of an alarm occurs, the alarm shall be generated. Alarm generation is simpler than fault propagation in that the alarm is always raised on the equipment on which the fault exists (that is, all alarm causality relationships are of the form CAUSES SAME).

The Event Generation function shall support OR-causality and AND-causality for alarms.

a. OR-caused alarms shall be generated when one or more of the causes of the alarm exist.

b. AND-caused alarms shall be generated when all of the causes of the alarm exist.

Alarms shall be time-stamped with the time of occurrence of the event or events that caused the alarm. In the case of OR-caused alarms, the time of the first-occurring cause shall be the time of the alarm. In the case of AND-caused alarms, the time of the last-occurring cause shall be the time of the alarm.

Event Removal

The Event Generation function shall remove events (both faults and alarms) for which the cause of the event has been removed (whether due to switchover (3.4.2.2.5) or due to removal by the user through the Simulation Input function (3.4.1)). Event removal shall be immediate when the cause of the event is removed.

a. For faults and OR-caused alarms with more than one cause, the event shall be removed when all of the causes have been removed.

b. For AND-caused alarms, the event shall be removed when any of the causes has been removed.

In order to perform event removal, the Event Generation function shall maintain, for every caused event, the primary key of every cause of that event. (The primary key is the fault (or alarm) symbol and the device, port, and side on which the fault or alarm occurred.)

Switchover

Certain equipment types known to TEG employ redundancy. These types may be identified by the presence of more than one entry in the Side table (3.4.5.3.1.3). In these types of equipment, only one redundant side is on-line at any time. Since only the on-line side is in communication with other equipment, fault propagation between equipment occurs only between the on-line sides. Therefore, switchover between sides will affect fault propagation, alarm generation, and event retraction. The Event Generation function shall execute automatic switchover and shall account for switchover in the constellation of generated events as follows:

a. When there is a fault on the on-line side of a device with redundancy and automatic switchover is enabled for that device, the Event Generation function shall:

   i. Make the current side of the device off-line.

   ii. Make the previously off-line side on-line.

   iii. Disable automatic switchover on that device until reenabled by the user.

b. When there is a switchover (whether automatic or manual), the Event Generation function shall:

   i. Remove all faults that had propagated from other devices to the previously on-line side and propagate these to the new on-line side.

   ii. Remove all faults that had propagated from the previously on-line side to other devices.

iii. Propagate all previously existing and newly propagated faults from the new on-line side within the device and to other devices.

Event Generation Function Outputs

The outputs of the Event Generation Function shall include the fault constellation and the alarm constellation.

D.3.4.3 Polling Simulation Function

The Polling Simulation Function shall simulate TRAMCON polling. TRAMCON polls one site and waits for the response, which takes approximately six (6) seconds, before polling the next site. If a response to a poll is not received within a ten-second time-out period, TRAMCON times out and continues polling the next site in the sequence.

Polling Simulation Function Inputs

The inputs to the Polling Simulation function shall include information from the Simulation Input function reflecting polling sequence and poll selections made by the user and faults and alarms output by the Event Generation function. This function shall also use information included in the Equipment, Connection, and Causality tables as inputs.

Polling Simulation Function Processing

The Polling Simulation Function shall simulate polling of each site included in the specified polling sequence. Polling shall occur in the same sequence as specified in the polling sequence. The starting time, number of times each site is polled, and polling interval shall be as specified in the polling selection. The poll selections made by the user are described as overall poll requests. This function shall interpret each overall poll request and convert it to individual specific poll requests. For example, an overall poll request specifying that each site is to be polled four times will be converted to four specific poll requests.

Changes in polling sequence or frequency shall take effect at the specified time unless polling of a site is in progress at that time, in which case the changes will take effect immediately following completion of polling at that site.

The poll time shall be incremented by six seconds each time another site is polled. This function shall detect a no response situation based on faults specified in the TRAMCON equipment. When a no response situation is detected, the poll time shall be incremented by ten seconds when the next site is polled.

This function shall examine the alarm constellation and shall select those alarms which would be received in response to an actual TRAMCON poll. Selection criteria for output shall include:

a. the fault responsible for the alarm must be in existence at the time the poll is made (i.e., must have already been inserted and not yet repaired).

b. the alarm must either be a Datalok 10 alarm that is transmitted to TRAMCON or an alarm derived by the TRAMCON system.

c. for Datalok 10 alarms, the TRAMCON equipment connecting the site at which the appropriate Datalok 10 device (i.e., the Datalok 10 which detects the fault that causes the alarm) is located to the site at which the TRAMCON Master is located must not contain any faults that would result in inability of either the poll request to be received at the Datalok 10 or the poll response to be received at the Master.

This function shall determine, for each poll handshake, whether or not the poll communication can be successful.

Polling Simulation Function Outputs

The outputs of the Polling Simulation function shall include a list of all TRAMCON alarms generated that correspond to a particular poll, in the form of poll responses. The output for each fault shall include complete information identifying the fault (e.g., equipment ID, port, side), the time at which the alarm was detected by the poll, and a textual description of the alarm.

D.3.4.4 Output Control Function

Output Control Function Inputs

The inputs to the Output Control function shall include valid requests from external programs (e.g., MITEC) and poll responses provided by the Polling Simulation function.

Output Control Function Processing

The Output Control function shall interpret requests, assemble the poll responses into the requested report, and output the requested report in accordance with the protocol specified in 3.3.3.1.

Output Control Function Outputs

The outputs of the Output Control function shall include a list of all TRAMCON faults and alarms generated from the most recent poll in a format that conforms to the TEG to MITEC CSCI interface requirements (paragraph 3.3.3.1). Each of these outputs shall be one of the following types, depending upon the request received from MITEC:

a. Summary - 1 line / polled site

b. Detailed - 1 line/alarm at site specified in MITEC request

Examples of summary and detailed reports are shown in Figures 7 and 8.

Future increments of TEG will support output of DPAS alarms.

D.3.4.5 Database Function

The TRAMCON Event Generator Knowledge Base defines the data tables used to store the representations of equipment, interconnections, circuits, faults, alarms, and site locations known to TEG.

The Knowledge Base defines both tables that are specific for each unique TRAMCON segment as well as tables that are applicable to all TRAMCON segments. These two sets will be known as the segment-specific knowledge base and the non-segment-specific knowledge base, respectively. Each separate segment-specific knowledge base and the non-segment-specific knowledge base shall be contained in separately loadable files. The first increment of TEG shall model the DEB IIA TRAMCON segment. Future increments of TEG shall support the capability to model any TRAMCON segment.

Database Function Inputs

For the first phase of TEG development, inputs to the database function shall be made through a text editor. Database function inputs for future phases of TEG are TBD.

Database Function Processing

Database processing in the first increment of TEG shall provide the functions to enter, edit, and delete individual facts and blocks of facts. If the rule language requires facts to be "loaded" or "compiled" the database function shall provide the requisite processing. Database function processing for future phases of TEG is TBD.

Database Function Outputs

The outputs of the Database Function shall include all tables that constitute the TEG Knowledge Base. The TEG Knowledge Base shall include the tables used to store the representations of equipment, interconnections, circuits, faults, alarms, and site locations known to TEG.

Equipment Representation

The following tables shall be used to define the representation of equipment known to TEG, aside from their interconnections: the Equipment, Port, Side, and Equipment Status tables. Additional tables required to define the attributes of a equipment may be specified at a later time.

Equipment Table

The Equipment table shall define the static attributes of each item of equipment known to TEG. The Equipment table shall be a part of the segment-specific knowledge base. Equipment shall include multiplexers, encryption equipment, and radios; this list may be extended as needed in the future. Equipment shall not include the transmission medium, e.g. wires,

patch panels, or radio links. Equipment table records may not be retracted by other functions of TEG. The Equipment table shall consist of one record for each item of equipment; each record will contain the fields defined in the subparagraphs below.

Equipment Table ID

This field will contain the symbol EQUIP. It shall identify all Equipment table records and distinguish these from all other types of record.

Equipment Class

This field shall contain a symbol that identifies the class of equipment represented by the record, e.g. FCC-99 or FRC-171.

Equipment ID

This field shall contain a symbol that identifies the item of equipment represented by the record. This symbol will be the primary key of the Equipment table; therefore, it will uniquely identify the item of equipment. In the simulation of a TRAMCON segment for which the equipment nomenclature is known, the symbol shall be the actual name of the item of equipment; otherwise, it shall be a meaningful and unique name.

Equipment Location

This field shall contain a symbol that identifies the facility at which the equipment is located. The set of symbols used in this field will be the set of three-letter facility IDs (e.g. DON for Donnersberg) used in the DEB.

Equipment Pretty Name

This field shall contain a string that describes the item of equipment. This string will be for use by display and report software that needs a suitable print name for the item.

Port Table

The Port table shall define the port symbols that are used for each class of equipment. The Port table shall be a part of the non-segment-specific knowledge base. There shall be at least one Port table record for each class of equipment, but most classes will have two or more records. Port table records may not be retracted by other functions of TEG. Each record of the Port table will consist of the fields defined in the subparagraphs below. Note that the primary key of the Port table consists of both the Equipment Class and the Port Symbol fields.

Port Table ID

This field will contain the symbol PORT. It shall identify all Port table records and distinguish these from all other types of record.

Equipment Class

This field shall contain a symbol that identifies the class of equipment that possesses the port defined by this record; for example, FRC-171. The set of symbols used in this field shall be the same as the set of symbols used in the Equipment Class field of the Equipment table (3.4.5.3.1.1.2).

Port Symbol

This field shall contain a symbol that is the name of a port on the subject class of equipment. The symbol FAR is distinguished: there shall be at most one 'far' port record for each class of equipment. Any symbol other than FAR is the name of a 'near' port. For example, the FRC-171 radio has the ports MBS-1, MBS-2, SCBS, and FAR.

Equipment that is always a data source or sink to the TRAMCON-monitored system shall have just one Port table record: the 'far' port record. The FCC-98 and CY-104A are examples of such equipment. Other equipment classes shall have two or more Port table records.

The numbering of 'near' ports shall follow the Tech Control practice for each class of equipment. When this is not known, the 'near' ports will be numbered from 1 through N, where N is the total number of 'near' ports. For non-multiplex equipment, such as the KG-81, the symbol for the 'near' port may be NEAR. The symbol NEAR shall not be used as a port symbol for multiplex equipment.

The convention for the distinction between 'near' and 'far' ports is that the 'near' ports are those conceptually 'closer' to the end user or the abstract 'center' of a through facility; the 'far' ports are those 'closer' to the transmission medium between facilities. The demultiplexed ports of a multiplexer are 'near'; the multiplexed port is 'far'. The unencrypted ('clear' or 'black') port of a KG-81 is 'near'; the encrypted ('red') port is 'far'. This distinction may not survive for matrix equipment such as DACS II and may need to be revisited when it becomes necessary to model such equipment.

Side Table

The Side table shall define the side symbols and automatic switchover attributes that are applicable for each class of equipment. The Side table shall be a part of the non-segment-specific knowledge base. There shall be at least one Side table record for each class of equipment, but some classes will have two records. Side table records may not be retracted by other functions of TEG. Each record of the Side table will consist of the fields defined in the subparagraphs below. Note that the primary key of the Side table consists of both the Equipment Class and the Side Symbol fields.

Side Table ID

This field will contain the symbol SIDE. It shall identify all Side table records and distinguish these from all other types of record.

Equipment Class

This field shall contain a symbol that identifies the class of equipment that possesses the side defined by this record; for example, FRC-171.

Side Symbol

This field shall contain a symbol that is the name of a side on the subject class of equipment. The set of legal symbols is: Only, A, B. Only will be used to represent classes of equipment that have only one side (i.e., no manufactured redundancy). For example, the FCC-98 second level multiplexer has Only one side. A and B will be used to represent classes of equipment possessing manufactured redundancy. For example, the FRC-171 radio has the sides A and B.

Switchover Attributes

This field shall contain a symbol that indicates the automatic switchover attribute for the subject class of equipment. The set of legal symbols is: ALWAYS, TOGGLE, UNKNOWN, NEVER. ALWAYS is used to indicate that automatic switchover is always enabled for the subject class of equipment. TOGGLE is used to indicate that the switchover state may be toggled, via operator input, between automatic and manual switchover. UNKNOWN is used to indicate that the switchover attribute is unknown. NEVER is used to indicate that switchover never occurs. This symbol should be used with all equipment that only has one side.

Equipment Status Table

The Equipment Status table shall define the modifiable attributes of each item of equipment known to TEG. Equipment shall include all items contained in the Equipment table. The Equipment Status table shall be a part of the segment-specific knowledge base. The Equipment Status table shall consist of one record for each item of equipment. Equipment Status records may be retracted by other TEG functions. Each record will contain the fields defined in the subparagraphs below.

Equipment Status Table ID

This field will contain the symbol EQSTATUS. It shall identify all Equipment Status table records and distinguish these from all other types of record.

Equipment ID

This field shall contain a symbol that identifies the item of equipment represented by the record. This symbol will be the primary key of the Equipment Status table; therefore, it will uniquely identify the item of equipment. The symbols shall match those used in the Equipment ID field of the Equipment table.

Operational Side

This field shall contain a symbol that identifies the side on which the equipment specified in the Equipment ID field is currently operating. The symbols shall match those used in the Side Symbol field of the Side table. Until contradictory information is provided, the initial operational side for redundant devices will be assigned the symbol A.

Automatic Switchover State

This field shall contain a symbol that indicates whether or not automatic switchover is currently enabled for the device specified in the Equipment ID field. The set of symbols will be: AS-ENAB, AS-DISAB, to represent auto-switchover enabled and auto-switchover disabled, respectively. Until contradictory information is provided, the initial automatic switchover state for all devices will be assigned the symbol AS-ENAB.

Connectivity Representation

The following tables shall be used to define the interconnections of equipment known to TEG: the Connection table and the Link table. Additional tables required to define attributes of connectivity may be specified at a later time.

Connection Table

The Connection table shall define the interconnections between items of equipment; it shall also provide a 'hook' for the future representation of patch panels, patches, and matrix switching. The Connection table shall be a part of the segment-specific knowledge base. There shall be at most two (and usually one) Connection table records for each pair of connected ports known to TEG; the Connection table record shall denote that a connection (whether in one or both directions) exists between these ports. This implies that for a duplex connection there will be one Connection Table record, and both the sender and receiver fields will denote ports capable both of sending and of receiving data. Two Connection table records may be used to denote a duplex connection in which the connection paths actually differ, if this should be necessary.

Each record of the Connection table will consist of the fields defined in the subparagraphs below. Note that the primary key of the Connection table consists of all of these fields: Connection Status, Sending Equipment ID, and Sending Port; the combination Connection Status, Receiving Equipment ID, and Receiving Port is also a candidate key and may be used as such.

The Connection table shall also identify ports that are either not connected or are connected to equipment not modeled in TEG. The Equipment ID symbols described under 3.4.5.3.2.1.4 shall be used for this purpose. Note that the use of one of these symbols as the Sending Equipment ID specifies the connection status of the corresponding Receiving Equipment ID and Port, while the use of one of these symbols as the Receiving Equipment ID specifies the connection status of the corresponding Sending Equipment ID and Port. It

would be meaningless for both the Sending and Receiving Equipment ID fields to contain one of these symbols.

Connection Table ID

This field will contain the symbol CONN. It shall identify all Connection table records and distinguish these from all other types of records.

Connection Status

This field shall contain a symbol to represent the status of the connection. The keywords for this field are NOMINAL and ACTUAL. A connection with 'nominal' connection status shall be the connection as specified in the TSO, multiplex plan, or other appropriate source of information as to the intended status of the connection in normal operation. A connection with 'actual' connection status shall be the connection as established by a patch, cross-connect, or other variation from the normal connection. The initial increment of TEG shall not be required to support the connection status of ACTUAL.

The concept of 'nominal' vs. 'actual' connection status serves to distinguish the original state of the network from a state that currently exists due to the presence of a patch. While patches may be used as temporary workarounds in operation, they are generally removed as soon as the failed equipment has been repaired and is placed back in service. Therefore it is necessary to retain the nominal state of the network in order to restore this state following the removal of a patch.

In database search, software that is concerned with the network as it is presently constituted (for example, in fault propagation) should first attempt to bind an 'actual' connection for a given sender or receiver; if this attempt fails, it should then attempt to bind a 'nominal' connection.

Duplexity

This field shall contain a symbol that denotes whether the connection is one-way (simplex) or two-way (duplex). The symbols SIMPLEX and DUPLEX will be used to represent simplex and duplex connections, respectively. In the case of a 'duplex' connection, the Sending Equipment ID and Sending Port fields may refer to either of the ports participating in the connection: the assignment of 'sender' and 'receiver' may be arbitrary. Consequently, software that is searching for the connection of a given port should be capable of selecting the connection record whether the given port is the 'sender' or the 'receiver'.

Sending Equipment ID

This field shall contain the Equipment ID symbol of the equipment participating as sender in the connection. The Equipment ID symbol will be either a unique Equipment ID as defined in 3.4.5.3.1.1.1 or any of the following keywords: SPARE, UNUSED, FAILED, or UNKNOWN. The initial

increment of TEG shall not be required to support the symbols SPARE, FAILED, or UNKNOWN. The interpretation of these keywords is as follows:

The use of SPARE as a Sending Equipment ID will identify a receiver that is designated for use as a spare in patching.

The use of UNUSED as a Sending Equipment ID will identify a receiver that is not used for any purpose (and presumably could be used as a spare). The distinction between SPARE and UNUSED is thought to be useful because the sparing and service restoration algorithms of MITEC make use of this distinction.

The use of FAILED as a Sending Equipment ID will identify a receiver that is not used because it has failed and has been removed from service. This is only possible for Connection records of ACTUAL status.

The use of UNKNOWN as a Sending Equipment ID will identify a receiver that is connected to equipment not represented in this database.

Sending Port

This field shall contain the Port symbol of the port participating as sender in the connection. The Port symbol shall be as defined above. When the Sending Equipment ID is any of the keywords SPARE, UNUSED, FAILED, or UNKNOWN, the Sending Port symbol will be the same as the Receiving Port symbol.

Receiving Equipment ID

This field shall contain the Equipment ID symbol of the equipment participating as receiver in the connection. The Equipment ID symbol shall be as defined above or may be any of the following keywords: SPARE, UNUSED, FAILED, or UNKNOWN. The interpretation of these keywords is as defined above except these symbols will here identify the receiver.

Receiving Port

This field shall contain the Port symbol of the port participating as receiver in the connection. The Port symbol shall be as defined above. When the Receiving Equipment ID is any of the keywords SPARE, UNUSED, FAILED, or UNKNOWN, the Receiving Port symbol will be the same as the Sending Port symbol.

Connection Medium

This field shall identify the type of connection that exists between the sending and receiving ports. The set of connection types that shall be supported by the initial increment of TEG shall be WIRE, MICRO, and NONE. The set of connection types that will be considered for support in future increments of TEG includes JACK, PATCH, TEST, TROPO, CABLE, SATCOM, and MATRIX. The matter of defining connection medium types is not settled, and considerable change in this area may be expected.

WIRE will be used to represent a hard-wired connection between the sending and receiving ports. In the initial increment of TEG, which will not support patching, this type shall subsume the types JACK, PATCH, and TEST.

MICRO will be used to represent a microwave radio connection between the sending and receiving ports.

NONE will be used to represent a port that is not connected to anything. Some ports that are actually disconnected or connected through a medium unknown to TEG may use this symbol.

JACK will be used to represent a connection made through a patch panel jack. JACK connections shall be restricted to those made through a set of jacks in their normal (i.e. closed) configuration; connections that involve an open jack shall be of type 'patch'.

PATCH will be used to represent a connection made through a patch panel using a patch cord or plug.

TEST will be used to represent a connection made through a test and access point.

TROPO will be used to represent a troposcatter radio connection between the sending and receiving ports.

CABLE will be used to represent a fiber-optic or wire land line or submarine cable connection between the sending and receiving ports.

SATCOM will be used to represent a communications satellite connection between the sending and receiving ports.
MATRIX will be used to represent a connection through a matrix switch (e.g. a DACS II) between the sending and receiving ports.

Connection ID

This field shall be a symbol that together with the Connection Medium symbol identifies the individual connection between the sender and receiver. The set of legal values for this symbol depends on the value of the Connection Medium symbol. With the exception of NONE, the Connection Medium and Connection ID fields, taken together, correspond to the primary key of the Link table.

For a Connection Medium value of NONE, the Connection ID will also be NONE.

For a Connection Medium value of WIRE, the Connection ID may be NONE, or it may be some symbol identifying the particular wire. If it is desired to simulate a fault involving the wire itself, the Connection ID should not be NONE. If not NONE, the symbol will correspond to a record in the Link table for that wire.

For a Connection Medium value of MICRO, the Connection ID will be a symbol identifying the particular microwave link. This symbol shall correspond to a record in the Link table for that microwave link.

Connection ID symbols for other values of Connection Medium are TBD.

Link Table

The Link table shall provide a 'hook' for simulating failures on transmission links and (in later increments of TEG) for implementing patches, cross-connects, and tests. The Link table shall be a part of the segment-specific knowledge base. Link table records may not be retracted by other TEG functions. This table will consist of the following fields:

Link Table ID

This field will contain the symbol LINK. It shall identify all Link table records and distinguish these from all other types of record.

Link Class

This field shall contain a Link Class symbol. This will be any of the symbols defined for Connection Medium (3.4.5.3.2.1.8), except NONE.

Link ID

This field shall contain a Link ID symbol. This shall correspond to a symbol used in a Connection ID field (3.4.5.3.2.1.9).

Link Pretty Name

This field shall contain a string that describes the specified link. Software that needs a print name for the link will use this field.

Circuit Representation

The following tables shall be used to define the representation of circuits known to the TEG: the Circuit table.

Circuit Table

The Circuit table shall define each circuit known to TEG. The Circuit table shall be a part of the segment-specific knowledge base. Since the connectivity of all equipment is fully specified by the Equipment, Port, Connection, and Link tables, the only additional information needed to define the circuit is the end points of the circuit. For future extensions of TEG, it may be necessary to define additional characteristics of the circuit that will be stored in this table. Circuit table records may not be retracted by other TEG functions. The Circuit table will consist of the following fields:

Circuit Table ID

This field will contain the symbol CKT. It shall identify all Circuit table records and distinguish these from all other types of record.

Circuit ID

This field shall contain a symbol that is the unique identifier of the circuit. This field will be the primary key of the Circuit table. For most circuits known to TEG, this will be the CCSD used in Tech Control.

Origin Equipment ID

This field shall contain a symbol that is the unique identifier of the equipment where the circuit originates. This symbol shall be an Equipment ID and correspond to a record in the Equipment table. (For a unidirectional, or simplex, circuit, the origin is the sending end. For a bidirectional, or duplex, circuit, the choice of origin is arbitrary; the preferred choice is the end that appears at the top or left-hand side of a multiplex plan or other appropriate source.) For trunk circuits, which are the only kind to be implemented in the initial increment of TEG, the Origin Equipment ID should be a multiplexer, the 'far' port of which is the origin of the trunk.

Origin Port

This field shall contain a symbol that identifies the port of the equipment where the circuit originates. This symbol shall be a Port symbol valid for the class of the origin equipment. For trunk circuits, the Origin Port should normally be FAR, specifying the 'far' port of the originating multiplexer.

Destination Equipment ID

This field shall contain a symbol that is the unique identifier of the equipment where the circuit terminates. This symbol shall be an Equipment ID and correspond to a record in the Equipment table. For trunk circuits, which are the only kind to be implemented in the initial increment of TEG, the Destination Equipment ID should be a multiplexer, the 'far' port of which is the destination of the trunk.

Destination Port

This field shall contain a symbol that identifies the port of the equipment where the circuit terminates. This symbol shall be a Port symbol valid for the class of the destination equipment. For trunk circuits, the Destination Port should normally be FAR, specifying the 'far' port of the destination multiplexer.

Circuit Pretty Name

This field shall contain a string that provides a description of the circuit. Software that needs a print name for the circuit should use this field.

## Fault Representation

The following tables shall be used to define the representation of faults known to TEG: the Event table and the Causality table. Additional tables required to define the attributes of faults may be specified at a later time.

### Event Table

The Event table shall define the attributes of each possible status, failure and alarm recognizable by TEG. The Event table shall be a part of the non-segment-specific knowledge base. Failures shall include both primary and secondary (also known as sympathetic) failures. Alarms shall include all alarms detectable by TEG resulting from primary and secondary failures. Event table records may not be retracted by other TEG functions. The Equipment Class and Event ID fields, taken together, correspond to the primary key of the Event table. The Event table shall consist of one record for each type of failure; the Event table shall consist of the following fields:

#### Event Table ID

This field will contain the symbol EVENT. It shall identify all Event table records and distinguish these from all other types of record.

#### Equipment Class

This field will contain a symbol that identifies the class of equipment associated with the event represented by the record. The set of symbols used in this field shall be the same as the set of symbols used in the Equipment Class field of the Equipment Table.

#### Event ID

This field shall contain a symbol that identifies the fault or alarm represented by the record, e.g. RX-FAILURE.

#### Event Source

This field shall contain a symbol that describes the source of the event. The set of symbols used in this field will be: PRIMARY, SECONDARY, DATALOK, TRAMCON, and DPAS. PRIMARY will be used to represent an actual failure. SECONDARY will be used to represent a failure resulting from and depending strictly upon a PRIMARY failure. DATALOK will be used to represent an alarm reportable by a Datalok-10 device. TRAMCON will be used to represent alarms derived by TRAMCON based on combinations of DATALOK alarms. DPAS will be used to represent alarms reportable by the DPAS.

#### Event Pretty String

This combination of fields will consist of zero or more String Component-Substitution ID pairs, followed by a Completing String. The string resulting from processing these fields will result in a single string that

will be used for reporting an event to interfacing units. The resulting string will also be for used by display and report software that needs a suitable print name for the item.

String Component-Substitution ID pairs

This pair of fields will consist of a string portion followed by a substitution ID portion. There may be zero or more Event Pretty String Component-Substitution ID pairs.

String Component

This field will contain a portion of a string that describes an event, including the aspect related to the corresponding substitution ID field. It will include any characters necessary for formatting.

Substitution ID

This field will contain a generic field name which will be used for the purpose of substituting specific names in the event string component. The set of symbols used in this field will be: ID, PORT, SIDE. ID will be used to represent a substitution for actual equipment ID in the event string component. PORT will be used to represent a substitution for actual port symbol in the event string component. SIDE will be used to represent a substitution for the actual side of the equipment associated with the event in the event string component.

Completing String

This field will contain the remaining portion of a string that describes an event. It will include characters necessary for formatting event time and any required overall string formatting characters.

Causality Table

The Causality table shall express the causal relations between faults and alarms. The Causality table shall be a part of the non-segment-specific knowledge base. All events whose source is not primary shall be represented in one or more records of the Causality table. Causality table records may not be retracted by other TEG functions. The primary key of the Causality table consists of all of these fields: Causing Event ID, Resulting Equipment Class, Resulting Event ID. The Causality table will consist of the following fields:

Causality Table ID

This field will contain the symbol CAUSES. It shall identify all Cause table records and distinguish these from all other types of record.

Equipment Relationship

This field shall contain a symbol that identifies the relationship between the equipment causing the fault and the equipment in which the sympathetic fault is caused. The set of symbols used in this field will be: SAME, DISTANT, SENDER, and LINK-END.

SAME will be used to refer to the same instance of equipment as the equipment causing the fault, i.e. equipment having the same equipment ID.

DISTANT will be used to refer to a piece of equipment of the same class that is connected through the far port of the equipment causing the fault. Other pieces of equipment may reside between the two pieces, but no piece of equipment residing between the device causing the fault and the distant device may be of the same class as these devices.

SENDER will be used to refer to a piece of equipment directly connected to the equipment that causes the fault and is a sender to that equipment (either as the left-hand-side of a SIMPLEX CONN relationship or as either side of a DUPLEX CONN relationship (3.4.5.3.2)). There is no requirement that the two pieces of equipment be collocated at the same site; only that a CONN relationship exist.

LINK-END will be used to refer to a piece of equipment residing at one of the ends of the connection path of which the equipment causing the fault is a part.

Causing Port Symbol

This field shall contain a symbol that identifies the port associated with the causing event on the equipment causing the fault or alarm. The set of symbols used in this field will be: ANY, NEAR, FAR.

ANY will be interpreted to mean that a fault of the type contained in the Causing Event field on any port of the equipment causing the fault is capable of causing the sympathetic fault or alarm.

NEAR will be interpreted to mean that a fault of the type contained in the Causing Event field on one of the near ports of the equipment causing the fault is capable of causing the sympathetic fault or alarm.

FAR will be interpreted to mean that a fault of the type contained in Causing Event on the far port of the equipment causing the fault is capable of causing the sympathetic fault or alarm.

Causing Event ID

This field shall contain a symbol that identifies the causing fault or alarm. The set of symbols used in this field shall be a subset of the set of symbols used in the Event ID field of the Event Table.

Resulting Equipment Class

This field shall contain a symbol that identifies the type of device in which the sympathetic fault or alarm is resulting. The set of symbols used in this field shall be the set of symbols used in the Equipment Class field of the Equipment Table.

Resulting Port Symbol

This field shall contain a symbol that identifies the port on the equipment in which the sympathetic fault or alarm is resulting. The set of symbols used in this field will be: ALL, NEAR, FAR.

ALL will be interpreted to mean that a fault of the type contained in the Causing Event field is capable of causing the Resulting Event in all ports of the resulting device.

NEAR will be interpreted to mean that a fault of the type contained in the Causing Event field on one of the near ports of the equipment causing the fault is capable of causing the Resulting Event in the corresponding near port of the resulting device. (Note: corresponding near ports have the same port identifier, e.g., MBS-2). FAR will be interpreted to mean that a fault of the type contained in Causing Event is capable of causing the Resulting Event in the far port of the resulting device.

Resulting Side

This field shall contain a symbol that identifies the side of the equipment in which the sympathetic fault or alarm is resulting. The set of symbols used in this field will be: ALL-SIDES, OPER-SIDE.

ALL-SIDES will be interpreted to mean that a fault of the type contained in the Causing Event field is capable of causing the Resulting Event on all sides of the resulting device.

OPER-SIDE will be interpreted to mean that a fault of the type contained in the Causing Event field is capable of causing the Resulting Event in the operational side of the resulting device.

Resulting Event ID

This field shall contain a symbol that identifies the resulting fault or alarm. The set of symbols used in this field shall be a subset of the set of symbols used in the Event ID field of the Event Table.

Segment Representation

The following tables shall be used to define the attributes of a segment known to TEG: the Site table. Additional tables required to define the attributes of a segment may be specified at a later time.

Site Table

The Site table shall define each site located in a particular TRAMCON segment known to TEG. The Site table shall be a part of the segment-specific knowledge base. Site table records may not be retracted by other TEG functions. The Site table will consist of the following fields:

Site Table ID

This field will contain the symbol SITE. It shall identify all Site table records and distinguish these from all other types of records.

Site Symbol

This field shall contain a symbol that identifies the facility at which the equipment is located. The set of symbols used in this field will be the set of three-letter facility IDs (e.g. DON for Donnersberg) used in the DEB. This symbol shall be the primary key of the Site table.

Site Pretty String

This field shall contain a string that describes the site location. The string should contain both the full name of the area as it is known to the DEB community, e.g., "Donnersberg", "Reese-Augsburg", etc. as well as the site symbol. This string shall be used for reporting an event to interfacing units. This string will also be for used by display and report software that needs a suitable print name for the item. The site pretty string will also contain any characters required by display and report software for formatting. An example of the site pretty string is:

"Donnersberg (DON)%n"

## 3.5 Adaptation Requirements

This section describes data that can be modified to change the scope of TEG operation within the prescribed limits.

### 3.5.1 System Environment

Adaptation of TEG to different host computers shall be supported by use of a device-independent rule-based language and a device-independent high order language.

### 3.5.2 System Parameters

This section is not applicable to this specification.

### 3.5.3 System Capacities

At a minimum, storage for the following values in the data base for each TRAMCON segment modelled shall be supported:

a. twenty-one (21) sites

b. 500 devices

c. 750 connections

## 3.6 Quality Factors

This section defines the quality factors that are applicable to TEG and describes how the applicable quality factors will be applied to TEG.

### 3.6.1 Correctness Requirements

The correctness quality factor is the extent to which the system satisfies the requirements defined in this Software Requirements Specification.

The correctness of TEG will be assured by frequent walkthroughs during the design process and by conducting rigorous testing.

### 3.6.2 Reliability Requirements

The reliability quality factor is the extent to which the system is expected to consistently perform its intended function.

The reliability of TEG will be assured by conducting rigorous testing.

### 3.6.3 Efficiency Requirements

The efficiency quality factor is a measure of the efficient use of the computing resource and memory by the system.

The rule-based language algorithm will provide reasonable efficiency. Programming standards for the rule-based language will enhance the efficiency of the search algorithm. Computing and memory usage risk areas will be prototyped early in system development and the results of the prototyping will be factored into the resulting implementation.

### 3.6.4 Integrity Requirements

This requirement is not applicable to this specification.

### 3.6.5 Usability Requirements

The usability quality factor is a measure of the effort required to learn and operate the system.

The usability of TEG will be ensured by designing the man-machine interface to be a user friendly and menu driven system. All messages to the user shall be self-explanatory.

### D.3.6.6 Maintainability Requirements

The maintainability quality factor is a measure of the effort required to locate and fix errors in the software.

The maintainability of TEG will be assured by adherence to the programming standards called out herein. In particular, the use of well structured HOL code, well formed rules, and extensive commenting will minimize the number of latent errors in the system and minimize the effort required to locate and fix the errors.

If the rule-based language used is compatible with an appropriate semantic checker, a semantic checker shall be used for style checking and generating cross-references. If CLIPS is used, the Cross Reference, Style, and Verification (CRSV) utility will be used for these purposes.

### D.3.6.7 Testability Requirements

The testability quality factor is a measure of the effort required to qualify that the software performs its intended functions. All TEG software requirements shall be testable. The testability of TEG shall be assured by the use of a well structured man-machine interface. Testability shall be further enhanced through the use of an ASCII-only interface with MITEC and use of standard transmission protocols.

### D.3.6.8 Flexibility Requirements

The flexibility quality factor is a measure of the effort required to enhance the operational software.

Use of a data-driven rule-based language is an important element in providing for flexibility of the TEG software. Because of this feature, substantial changes may be made to the system by introducing changes into the data base, rather than modifying source code. Use of structured, modular coding within the high order language and well-formed rules in the rule-based language further enhance TEG's flexibility.

Future increments of TEG may include a graphical user interface. To the maximum extent possible, code developed for this purpose to support MITEC will be reused.

### D.3.6.9 Portability Requirements

The portability quality factor is a measure of the effort required to transfer the software from one hardware configuration and/or system environment to another.

The portability of TEG will be maximized by the use of portable rule-based and high order language and the avoidance of system-dependent features. When system-dependent features are required, those areas of the code will be isolated and identified with potential portability impacts.

D.3.6.10 Reusability Requirements

The reusability quality factor is a measure of the effort required to use the software in other applications.

Source code and data developed for use in TEG may be usable for the MITEC alarm filtering function and/or a future TRAMCON trainer.

D.3.6.11 Interoperability Requirements

The interoperability quality factor is a measure of the effectiveness of the system's interface with other systems.

TEG will be designed to communicate with MITEC in a manner consistent with the way TRAMCON and DPAS would communicate with MITEC if such communication links were provided.

D.4. QUALIFICATION REQUIREMENTS

D.4.1 General Qualification Requirements

The purpose of these qualification tests is to verify that TEG fulfills the requirements of this SRS. In order to assure the timely completion of system integration and acceptance test, qualification tests are begun during software development. In testing TEG, the evaluation of the following quality factors shall predominate:

a. Usability. Can the prospective user community of TEG operate the program and interpret its results without difficulty?

b. Correctness. Are the fault and alarm events generated by TEG the same as those that occur in TRAMCON: do the results of TEG correspond to known TRAMCON results, or are TEG results believable to experienced TRAMCON operators?

D.4.1.1 Qualification Approach

D.4.1.2 Qualification Phases

Qualification testing of TEG will be accomplished in two phases: Computer Program Test and Evaluation (CPT&E) and CSCI Acceptance Test (CSAT). The purposes of these two phases differ in that CPT&E is intended to verify primarily the correct operation of code modules and internal interfaces while CSAT is intended to verify the quality factors of usability and correctness discussed in 4.1 above.

Computer Program Test and Evaluation (CPT&E)

The first phase of TEG qualification testing will be CPT&E. CPT&E will be conducted by the TEG programming staff according to internally developed procedures. CPT&E will normally be used to verify internal and developmental requirements, such as the correct operation of individual code modules and

internal interfaces, software integration, and observance of design and programming standards. CPT&E will be conducted concurrently with programming and continue until TEG is complete and ready for CSAT. When CPT&E is used to verify a requirement stated in this SRS, the procedures and test results will be recorded and made available for inspection at CSAT.

CSCI Acceptance Test (CSAT)

The second and final phase of TEG qualification testing will be CSAT. CSAT will be conducted by the TEG programming staff, with witnesses from Lincoln Laboratory and the TRAMCON user community. Certain tests that demonstrate usability of TEG functions will be conducted by Lincoln Laboratory personnel and TRAMCON users. Test results that require expert evaluation will be evaluated by TRAMCON users, assisted by the TEG programming staff.

Problems detected during CSAT will be recorded and tracked until corrected or otherwise closed. The Lincoln Laboratory Program Manager shall be the final authority on the disposition of problem reports.

D.4.1.3 Qualification Methods

The following qualification methods shall be used to test TEG: inspection, analysis, demonstration, and test.

Inspection

Inspection is a form of testing in which a requirement is verified by reading off evidence that the requirement has been implemented. Inspection is most often used to verify that design and coding standards have been followed, or that required items are in fact present. For example, it may be verified by inspection that TEG stores the data needed to produce each of its required displays.

Analysis

Analysis is a form of testing that differs from inspection in that the evidence verifying a requirement cannot simply be read off, but must be argued or deduced. Analysis is commonly used to verify requirements that are otherwise impractical to test; for example, to predict the performance of software with a database much larger than that constructed for the test.

Demonstration

Demonstration is a form of testing in which a requirement is verified by executing the system and reading off some result that is evidence that the requirement has been met; for example, that TEG produces a display that contains the expected data. While demonstration is in general the preferred form for testing, it is not practical to use for all kinds of requirements.

Test

Test is a form of testing in which a requirement is verified by executing the system and analyzing the results in order to determine whether the requirement has been met. In this sense, test is a combination of demonstration and analysis. Certain kinds of requirements are susceptible to test but not to demonstration; for example, the requirement that TEG generate the correct set of alarms corresponding to a particular inserted fault, or that the alarm messages are meaningful to a TRAMCON operator.

## D.6. GLOSSARY

| | |
|---|---|
| AB | Air Base |
| AFB | Air Force Base |
| ASCII | American Standard Code for Information Interchange |
| AT&T | American Telephone & Telegraph |
| CCSD | Command/Control Service Designator |
| CCSO | Command and Control Systems Office |
| CLIPS | C Language Inference Programming System |
| CPT&E | Computer Program Test and Evaluation |
| CRSV | Cross Reference, Style, and Verification |
| CSAT | CSCI Acceptance Test |
| CSCI | Computer Software Configuration Item |
| DACS | Digital Access and Cross-Connect System |
| DCS | Defense Communications System |
| DEB | Digital European Backbone |
| DPAS | Digital Patch and Access System |
| DSOM | Digital Systems Operations Manual |
| HOL | High-Order Language |
| HWCI | Hardware Configuration Item |
| IBM | International Business Machines |
| ID | Identification |
| MBS | Mission Bit Stream |
| M.I.T. | Massachusetts Institute of Technology |
| MITEC | Machine Intelligent Technical Controller |
| NCS | Network Control System |
| NMES | Network Management Expert System |
| PC/AT | Personal Computer/Advanced Technology |
| PC-DOS | Personal Computer Disk Operating System |
| RDBMS | Relational Data Base Management System |
| RS-232 | EIA Recommended Standard 232, specifies electrical characteristics of serial connections. |
| SCBS | Service Channel Bit Stream |
| SQL | Structured Query Language |
| SRS | Software Requirements Specification |
| TBD | To Be Determined |
| TEG | TRAMCON Event Generator |
| TM | TRAMCON Master |
| TRAMCON | Transmission Monitor and Control |
| TSO | Telecommunications Service Order |
| XON-XOFF | Transmission On - Transmission Off |
| USAF | United States Air Force |

## Appendix E Pentagon - Ft. Detrick Demonstration Plans

The essential elements of the MITEC demonstrations are:

1. Application of modern remotely-controllable communications and test equipment

2. Autonomous fault isolation, circuit restoral by expert systems
   - Automatic response to alarms
   - Induced response to user complaints
   - Restoration via spares where available
   - Instructions to the operator on how to restore, if no spares

3. Automatic performance testing of VF circuits

The seven specific tests planned for the TCAPS demo are as follows:

TEST 1 OBJECTIVE: Demonstrate automatic response to an FCC-100 aggregate alarm.

TEST 1 METHODOLOGY: Induce a failure in the FCC-100 aggregate circuit at either site, by unseating the 56 Kbps channel bank card currently carrying the aggregate.

TEST 1 EVALUATION: Observe that the MITEC at the affected site receives an aggregate loss alarm, and that the two MITECs restore the circuit via switching to a spare channel bank card.

TEST 2 OBJECTIVE: Demonstrate automatic response to a T1 failure alarm from a channel bank.

TEST 2 METHODOLOGY: Induce a T1 failure by disconnecting it from the channel bank at either site.

TEST 2 EVALUATION: Observe that the MITEC at each site receives a carrier loss alarm and provides instructions for restoring T1 service (no spare T1 is provided).

TEST 3 OBJECTIVE: Demonstrate automatic resolution of a remote data circuit user complaint (alternative 1).

TEST 3 METHODOLOGY: Induce a failure in the end-user-to-end-user data circuit by unseating the FCC-100 port card serving that circuit, at either site. Observe signal loss light on the BERT acting as the user, and notify the local MITEC (by operator entry) of the corresponding user complaint.

TEST 3 EVALUATION: Observe that the MITECs automatically restore the circuit by switching to a spare FCC-100 channel.

TEST 4 OBJECTIVE: Demonstrate automatic resolution of a remote data circuit user complaint (alternative 2).

TEST 4 METHODOLOGY: Induce a failure in the end-user-to-end-user data circuit by disabling the 2.4 kbps modem serving that circuit, at either site. Observe signal loss light on the BERT acting as the user, and notify the local MITEC (by operator entry) of the corresponding user complaint.

TEST 4 EVALUATION: Observe that the MITECs automatically restore the circuit by switching to a spare modem.

TEST 5 OBJECTIVE: Demonstrate automatic resolution of a local data circuit user complaint (using VF channel in channel bank).

TEST 5 METHODOLOGY: Induce a failure in the local-user-to-local-user data circuit by unseating the channel bank VF card serving that circuit, at either site. Observe signal loss light on the BERT acting as the user, and notify the local MITEC (by operator entry) of the corresponding user complaint.

TEST 5 EVALUATION: Observe that the MITECs automatically restore the circuit by switching to a spare VF card in the channel bank.

TEST 6 OBJECTIVE: Demonstrate automatic resolution of a user VF tail circuit problem.

TEST 6 METHODOLOGY: Induce a VF tail circuit failure at either site by introducing excess loss through the phone line simulator. Observe signal loss light on the BERT acting as the user, and notify the local MITEC (by operator entry) that a user complaint has been received.

TEST 6 EVALUATION: Observe that the MITECs isolate the fault by resorting to loopback through the remote line unit. Observe that the MITEC at the affected site provides instructions for setting up a telco service call.

TEST 7 OBJECTIVE: Demonstrate automatic performance testing of a VF tail circuit.

TEST 7 METHODOLOGY: Log the VF tail circuit out at either site, and notify MITEC to conduct performance tests on it. At intervals, induce degradation in the circuit by means of the phone line simulator.

TEST 7 EVALUATION: Observe that the MITEC tests line performance by means of the RLU and the Hekimian 3701, correctly measuring the degradation introduced by the phone line simulator.

## REFERENCES


1. H.M. Heggestad, "Knowledge-Based Operation and Management of Communications Systems", SOAR '88, Dayton, OH, July 1988

2. Annual Report, "Knowledge-Based Systems Analysis and Control", MIT Lincoln Laboratory, 30 September 1986 (ESD-TR-87-041, AD A188 163)

3. Heggestad, H.M., "Dedicated Circuits Network Survivability Enhancement via Expert System Techniques", ICC '87, Seattle, WA, June 1987.

4. Otis, B.W., and Heggestad, H.M., "The Expert Tech Controller: A Network Control Expert System", MILCOM '87, Washington, DC, October 1987

# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*



| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE | 3. REPORT TYPE AND DATES COVERED |
|---|---|---|
| | 30 September 1989 | Annual Report, 1 October 1988 - 30 September 1989 |

4. TITLE AND SUBTITLE

Knowledge-Based System Analysis and Control

5. FUNDING NUMBERS

C — F19628-90-C-0002
PE — 33126F
PR — 295

6. AUTHOR(S)

Harold M. Heggestad

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lincoln Laboratory, MIT
P.O. Box 73
Lexington, MA 02173-9108

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

RADC/DCLD
Griffiss AFB, NY 13441

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ESD-TR-90-103

11. SUPPLEMENTARY NOTES

None

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The Machine Intelligent Tech Controller (MITEC) is a software system that operates upon a hardware testbed. The latter is a set of communications, switching and test equipment representing the state of the art as typically seen in modern commercial communications centers. As such these equipment items are typical of new equipment that would be bought and installed to modernize Tech Control Facilities (TCFs) throughout the Air Force if funds were budgeted for this purpose. In particular, they are remotely controllable from a computer or terminal. In the MITEC testbed all the equipment control lines are connected to the MITEC computer, and the MITEC software performs circuit fault isolation and service restoral by directly controlling the equipment, without human intervention.

Two MITEC systems are in place and undergoing further development at Lincoln Laboratory. In operation the two expert systems collaborate with each other much as skilled human Tech Controllers do: the first one to become aaware of a circuit fault initiates the diagnosis, while the other provides test signals and measurements upon request. Should the first MITEC determine that the fault is actually on the other MITEC's premises, control will be handed over and the second system will proceed with the diagnosis. The design and operation of the MITEC software and hardware is described in the body of the report.

Plans are developing to deploy two additional MITEC systems in the Washington area for demonstration purposes in the coming year. Another recent extension is commencement of preparations for TRAMCON (Transmission Monitoring and Control) system alarm interpretation and correlation. Work in both of these areas is described.

14. SUBJECT TERMS

expert systems
tech control
tech control automation
telecommunications
machine intelligence
Defense Communications System

15. NUMBER OF PAGES
166

16. PRICE CODE

| 17. SECURITY CLASSIFICATION OF REPORT | 18. SECURITY CLASSIFICATION OF THIS PAGE | 19. SECURITY CLASSIFICATION OF ABSTRACT | 20. LIMITATION OF ABSTRACT |
|---|---|---|---|
| Unclassified | Unclassified | Unclassified | SAR |